Automatic Dense 3D Scene Mapping from Non-overlapping Passive Visual Sensors for Future Autonomous Systems

HAMILTON, OLIVER

Automatic Dense 3D Scene Mapping from Non-overlapping Passive Visual Sensors for Future Autonomous Systems

Oliver K. Hamilton

Supervised by: Dr Toby P. Breckon

Thesis presented for the degree of
Doctor of Philosophy

School of Engineering & Computing Sciences
Durham University
Durham
England

2017

Abstract

The ever increasing demand for higher levels of autonomy for robots and vehicles means there is an ever greater need for such systems to be aware of their surroundings. Whilst solutions already exist for creating 3D scene maps, many are based on active scanning devices such as laser scanners and depth cameras that are either expensive, unwieldy, or do not function well under certain environmental conditions. As a result passive cameras are a favoured sensor due their low cost, small size, and ability to work in a range of lighting conditions.

In this work we address some of the remaining research challenges within the problem of 3D mapping around a moving platform. We utilise prior work in dense stereo imaging, Stereo Visual Odometry (SVO) and extend Structure from Motion (SfM) to create a pipeline optimised for on vehicle sensing.

Using forward facing stereo cameras, we use state of the art SVO and dense stereo techniques to map the scene in front of the vehicle. With significant amounts of prior research in dense stereo, we addressed the issue of selecting an appropriate method by creating a novel evaluation technique. Visual 3D mapping of dynamic scenes from a moving platform result in duplicated scene objects. We extend the prior work on mapping by introducing a generalized dynamic object removal process. Unlike other approaches that rely on computationally expensive segmentation or detection, our method utilises existing data from the mapping stage and the findings from our dense stereo evaluation. We introduce a new SfM approach that exploits our platform motion to create a novel dense mapping process that exceeds the 3D data generation rate of state of the art alternatives. Finally, we combine dense stereo, SVO, and our SfM approach to automatically align point clouds from non-overlapping views to create a rotational and scale consistent global 3D model.

Contents

Abstract **i**

Abbreviations **xvii**

Declaration **xviii**

Acknowledgements **xx**

1 Introduction **1**

1.1 Motivation 2

1.2 Related efforts 3

1.3 Summary 4

1.4 Thesis Contributions 6

1.4.1 Publications 6

1.5 Thesis Structure 7

2 Literature Review **8**

2.1 Stereo Vision 9

2.2 Visual Odometry 13

2.2.1 Review of Existing Approaches 14

2.3 Structure from Motion 17

2.3.1 Review of Existing SfM Approaches 18

2.4 3D Reconstruction 23

2.4.1 Review of Existing 3D Reconstruction Approaches 24

2.5 Dynamic Object Removal 26

2.6 Summary 28

3 Stereo Vision for 3D Mapping 30

3.1 Dense Stereo Imaging . . . 30

3.1.1 Matching Strategies . . . 31

3.1.2 Matching Cost . . . 32

3.1.3 Disparity Measurement . . . 33

3.1.4 Post Processing . . . 33

3.2 Quantitative Assessment of Dense Stereo Vision . . . 34

3.3 Results . . . 40

3.4 Conclusion . . . 45

3.5 Future Work . . . 47

3.6 Summary . . . 47

4 Structure from Motion 48

4.1 Review . . . 48

4.1.1 Typical SfM Process . . . 49

4.2 Proposed SfM Process . . . 52

4.2.1 Data Collection . . . 53

4.2.2 Hardware Configuration . . . 53

4.2.3 Data Capture . . . 54

4.2.4 Sparse SfM Using Optical Flow . . . 55

4.2.4.1 Features . . . 56

4.2.4.2 Matching . . . 57

4.2.4.3 Motion Extraction . . . 60

4.2.4.4 Triangulation . . . 62

4.2.4.5 Refinement . . . 64

4.2.4.6 Pre-Bundle Adjustment Point Filtering . . . 67

4.2.4.7 Dynamic Bundle Adjustment Window Size . . . 71

4.2.5 SfM Densification using Dense Stereo from Motion . . . 82

4.3 Summary . . . 88

5 Moving Object Removal 89

5.1 Introduction . . . 89

5.2 Process Overview 90
5.2.1 Visual Odometry 92
5.2.2 Scene Mapping 93
5.3 Dynamic Object Removal 93
5.3.1 Disparity Projections 93
5.3.2 Optical Flow From Disparity 96
5.4 Results 100
5.5 Conclusion 105
5.6 Summary 106

6 Large Scale Reconstruction and Evaluation 108
6.1 Introduction 108
6.2 Processing 109
6.2.1 Automatic Alignment 109
6.2.2 Data Log Repair 112
6.3 Results 117
6.4 Evaluation 119
6.5 Summary 124

7 Conclusions and Discussion 125
7.1 Conclusions 125
7.2 Contributions 126
7.2.1 Industrial Impact 127
7.2.2 Limitations 128
7.3 Further Work 129
7.3.1 Data Capture 130
7.3.2 Dense Stereo 130
7.3.3 Structure from Motion Pipeline 130
7.3.4 Extend Coverage 131

Bibliography 132

A Supplementary Material **141**

A.1 Further Stereo Results 141

A.2 Further Dense Stereo Range Angle Maps 144

A.3 Optical Flow Optimisaton Results 146

A.4 Dynamic Bundle Adjustment Windows Size and Filter Strategy Results 148

A.5 Evolution of Data Capture Platform 156

List of Figures

1.1 Google Earth [1] image showing 3D reconstruction of suburban housing in Milton Keynes, UK. 5

2.1 Example of KITTI laser scanner data projected into the left greyscale-camera coordinate system and scaled to simulate a disparity map. . . 12

2.2 Number of results submitted to Middlebury evaluation (version 2 and 3) by publication year *(Note: May contain duplicate publications as some are submitted twice with different parameters).* 12

2.3 Illustration of feature tracking across stereo cameras in motion in order to extract scaled odometry. 15

2.4 Photoscan [2] reconstruction using two input images. Top Left: Sparse point cloud from feature points used in the SfM process. Top Right: Dense point cloud from multi-view stereo process. Bottom Left: Mesh constructed from dense point cloud with mesh triangles naively coloured and lit. Bottom Right: Mesh with correctly UV mapped texture and calculated mesh normals to allow for correct lighting. . . 24

3.1 Sample of the input data (frame 17) from the left stereo camera in the KITTI dataset. 34

3.2 Example disparity maps of frame 17 from KITTI dataset from sequence 2011_09_26_drive_0009. (a) BM, (b) Cross, (c) SGBM, (d) AdaptDP, (e) NoMD, (f) ELAS. 34

3.3 Visualisation of colour mapped disparity point cloud and laser ground truth point cloud (white points). 36

3.4 Colour disparity point cloud generated using [3] with laser scanner point cloud (white dots) and annotated ground truth bounding boxes (red boxes). 37
3.5 Flow chart for performing quantitative evaluation of dense stereo algorithms on foreground objects of interest. 37
3.6 Raw ground truth data for a car at a range of 25m (black points). Ground truth data post ICP registration with disparity point cloud (red points). 39
3.7 Short range accuracy difference between max disparity parameter settings. Left: Raw data. Right: 0.5m smoothing window. 40
3.8 Top left: Stereo algorithms with higher accuracy. Top right: Less accurate algorithms in this study. Bottom left and right images have a 0.5m smoothing window applied. 42
3.9 The Range-Angle map showing a given stereo algorithm's accuracy coverage of an observed scene. ICP error is clamped to a maximum of 1m to enforce colour scale consistency across all range-angle maps. 43
3.10 Predicted stereo accuracy as function of range and azimuth for KITTI camera configuration. Left: Range error with 0.1px disparity error. Right: Range error with 0.2px disparity error. 44
3.11 An overlay of the stereo accuracy results on top of the stereo accuracy prediction with a given disparity matching error. Left: Dense stereo algorithm 'Cross' results overlaid on theoretical stereo matching error of 0.1px. Right: SGBM results overlaid on theoretical stereo matching error of 0.2px. 44

4.1 Illustration of epipolar geometry and how a point observed in two images can be related using the Fundamental Matrix, F. 50
4.2 Data logging pod mounted on our Mitsubishi i-MiEV electric vehicle. Left, schematic view from above. Right, Parked near Durham Cathedral, the site of many of our data collection activities. 55
4.3 Non-uniform distribution of SURF [4] feature detector points. Every 2^{nd} point drawn for viewing clarrity. 57

4.4 Examples of one-to-many feature matching using SURF descriptors. . 58
4.5 Features tracked using sparse optical flow. Black dot indicates the feature point start location, Red line indicates direction and magnitude of motion. 59
4.6 Results from image pair 1 in Figure 4.8. Left: Mean reprojection error in pixels (to 2 decimal places) after BA. Right: Number of valid tracked features used in BA. 60
4.7 Median values from all test cases. Left: Mean reprojection error in pixels. Right: Number of valid tracked features. 60
4.8 Heatmaps for empirical OF parameter optimisation. Each heatmap column has normalised scaling for illustrative comparison. 61
4.9 Projection of 2D points x and x' into 3D space point X. 63
4.10 Optical flow historic lookup. Red feature index shows features that failed to track to the next frame. Black are valid feature indexes. Dot-shaded cells indicate new features in a given frame. Blue shaded cells indicate flow history of example features. Green arrow indicates flow matches from frame to frame. 65
4.11 Flow features tracked over three frames and indexed using dynamic lookup tables. 66
4.12 Adaptive bundle adjustment window scheme. Circles represent features, columns represent frames. 66
4.13 Sparse point clouds output from pre-BA filter tests. Top to bottom, sequence 1 to 6 respectively. Left to right, filter strategies F1 to F3 respectively. 70
4.14 Bundle adjustment processing time as a function of number of features for different temporal window sizes and filtering strategies for sequence 5. 73
4.15 Reprojection error post BA stage of the SfM process for different sized temporal windows for sequence 5. 74
4.16 Dynamic window sizes used for our bundle adjustment approach of sequence 5. 75

4.17 Bundle adjustment processing time as a function of number of features for different temporal window sizes for sequence 2. 76
4.18 Reprojection error post BA stage of the SfM process for different sized temporal windows for sequence 2. 77
4.19 Dynamic window sizes used for our bundle adjustment approach of sequence 2. 78
4.20 Bundle adjustment strategy test results from sequence 3 (Section 4.2.4.6). 79
4.21 Reprojection error post BA stage of the SfM process for different sized temporal windows for sequence 3 (Section 4.2.4.6). 80
4.22 Dynamic window sizes used for our bundle adjustment approach of sequence 3. 81
4.23 Left: Disparity map created from rectified image pairs using SGBM [5] dense stereo matching approach. Right: Colour-coded depth map (blue = nearest, red = farthest) viewed from a different, virtual, camera position. 82
4.24 Dense raw point cloud reconstruction from SfM process using SGBM [5] for densification. 83
4.25 Reconstruction of a short image sequence viewed from a novel camera position high above the scene. Left, sparse points only. Middle, addition of dense reconstruction showing good structural agreement with sparse points. Right, middle point cloud viewed from a different viewpoint near ground level. 84
4.26 Three views of dense reconstruction around a corner from a left-hand turn. 85
4.27 Example sequential images from single side facing camera and the densification process that produces almost pixel wise dense depth maps. Top left and right, are frames n and $n+1$ respectively post rectification. Bottom left is an overlay of each rectified image. Bottom right, shows the densification using SGBM [5] 86

4.28 Dense reconstruction from our Durham, UK dataset viewed from two different virtual viewpoints to illustrate the 3D nature of the scene. . 87

4.29 Top view of Figure 4.28 showing different baselines for SfM densification. Left to right, densification performed using frames n to $n+1$, n to $n+2$, and n to $n+3$ respectively. 87

5.1 Main processing overview for moving object detection from a moving stereo camera platform. The naming convention in the diagram is as follows, ${}_{H}M_{T}^{V}$ – H is the handedness of the frame (Left or Right), M is the image or matrix, T is the time it originates from, V is the viewing time it is projected into. Hence, ${}_{L}S_{1}^{0}$ is the left stereo intensity image from $t = 1$ remapped into $t = 0$. S - Stereo Intensity Image; D - Disparity Map; OFFD - Optical Flow From Disparity; DMM - Disparity Moving Mask; IMM - Intensity Moving Mask; SVO - Stereo Visual Odometry; RP - Reprojection Calculation; DD - Disparity Difference; ID - Intensity Difference. 91

5.2 Left: A car driving forwards, ahead of the logging platform, reconstructed multiple times cluttering up the global point cloud. Right: Moving object removed from the final reconstruction. 92

5.3 Top: Disparity map at t. Middle: Disparity map at $t+1$. Bottom: Disparity map at $t+1$ back projected into t. Vertical red lines illustrate the alignment between scene features at t, $t+1$ and $t+1$ back projected to t. 95

5.4 Left: Intensity image from left stereo camera. Right: Threshold of disparity difference image or DMM. 96

5.5 Flow from disparity of the illustrated failure mode in Figure 5.4 with the vehicle driving forwards and right (flow direction and magnitude are represented respectively by the hue and value channels of the HSV colour space). 98

5.6 Top: Intensity image at time t. Middle: Intensity image at $t+1$. Bottom: Intensity image at $t+1$ projected to t. Vertical red lines illustrate alignment of various features, demonstrating the reprojection accuracy. The white region on the left of the bottom image, this is unmatched region of the disparity map corresponding to the maximal disparity search window. 99

5.7 Disparity maps with moving objects masked out. Shadows cast by the moving objects are also masked out due to intensity image variation. 100

5.8 Left to Right: Two different viewpoints of the same point cloud. Top: Moving object reconstructed multiple times. Bottom: Moving object masked out of the reconstruction process. 102

5.9 Top: A slow moving pedestrian with approximately 50% self-overlap between frames. Bottom: Successfully removed from mapping solution. 103

5.10 Top: Large group of people moving at various speeds including some static bystanders. Bottom: Pedestrians are largely removed, elements that remained static between frames are still present. 104

5.11 Top: Fast moving pedestrian with motion perpendicular to camera motion. Bottom: Successful removal of moving components, however the feet remain visible in the point cloud as these are temporarily static during contact with the ground. The motion of the leg above the ankle has been successfully removed. 105

6.1 Pose graph alignment with camera mounting plate overlay for illustration presented in the stereo camera coordinate system. 112

6.2 SVO performance in the presence of randomly dropped frames. Thick black line represents results from no dropped frames. Black circles indicate end positions of SVO runs. *Note: Axes are not equal scales.* . 114

6.3 Top: Real frame (n) from left stereo camera at time T. Middle: Synthetic left stereo images at approximately $T+dt$. Bottom: Real frame ($n+1$) from left stereo camera at time $T+2dt$. 115

6.4 Left: Plot showing the different odometry paths with and without frame repair by image synthesis. Right: Overlay of camera paths on approximate location demonstrating the impact on positional error, map underlay from [1]. 116

6.5 Top: Image from Google Earth [1]. Middle: Point cloud from SfM reconstruction viewed using Meshlab [6] (9.7 million points). Bottom: Same location captured an hour later with more stationary pedestrians present (11.2 million points). 117

6.7 Orthographic projection of point cloud from North Bailey sequence (21.5 million points). *Note: this is a raw point cloud with no meshing or texturing.* 118

6.6 A close up render of the same point cloud in Figure 6.5 of the region highlighted with the red square, demonstrating the fine detail of the reconstruction. 118

6.8 Reconstruction from South Bailey, Durham, UK. 119

6.9 SfM densification showing fine repeating structure of railings being accurately reconstructed. 119

6.10 Top: Orthographic projection of dense SfM reconstruction. Bottom: Perspective projection of same point cloud from a different virtual camera position. 120

6.11 Combination point cloud created from dense SfM (Chapter 4) and dense stereo [5]. 121

6.12 Top: Google Street View [7] image showing the North Bailey location. Bottom: Reconstruction from 3 non-overlapping views auto-aligned. 121

6.13 Left: Ground level view of aligned reconstruction. Right: Birdseye view of the same combined point cloud. 122

6.14 Top, partial reconstruction of Durham Marketplace using our SfM approach. Bottom, same sequence reconstructed using a leading commercial package [2] 124

A.1 Dense stereo disparity maps generated using the KITTI data. Algorithms used, top to bottom: BM, SGBM, NoMD, Cross, AdaptDP, ELAS. 142
A.2 Dense stereo disparity maps generated using the KITTI data. Algorithms used, top to bottom: BM, SGBM, NoMD, Cross, AdaptDP, ELAS. 143
A.3 Left and middle: Stereo input images from our data set. Right: Dense disparity map created with SGBM. 144
A.4 Range-angle maps showing accuracy of dense stereo algorithms from KITTI image sequence 2011_09_26_drive_0009 using process outlined in Chapter 3. 145
A.5 Range-angle maps showing accuracy of dense stereo algorithms from KITTI image sequence 2011_09_26_drive_0023 using process outlined in Chapter 3. 146
A.6 Full results for the charts in Figure 4.8. 147
A.7 Bundle adjustment processing times for different window sizes and filtering strategies. Sequence 1 in Section 4.2.4.7. 148
A.8 Reprojection error post BA stage of the SfM process for different sized temporal windows and filter strategies. Sequence 1 in Section 4.2.4.7. 149
A.9 Dynamic window sizes used for our bundle adjustment approach of Sequence 1 in Section 4.2.4.7. 150
A.10 Bundle adjustment processing times for different window sizes and filtering strategies. Sequence 4 in Section 4.2.4.7. 151
A.11 Reprojection error post BA stage of the SfM process for different sized temporal windows and filter strategies. Sequence 4 in Section 4.2.4.7. 152
A.12 Dynamic window sizes used for our bundle adjustment approach of Sequence 4 in Section 4.2.4.7. 153
A.13 Bundle adjustment processing times for different window sizes and filtering strategies. Sequence 6 in Section 4.2.4.7. 154
A.14 Reprojection error post BA stage of the SfM process for different sized temporal windows and filter strategies. Sequence 6 in Section 4.2.4.7. 155

A.15 Dynamic window sizes used for our bundle adjustment approach of Sequence 6 in Section 4.2.4.7. 156

A.16 Initial robot platform based configuration. Hardware components labelled. The short wheel base caused excessive roll and pitch when traversing even slightly rough ground, this manifested as image artifacts during the camera integration period due to a rolling shutter based CCD sensor. 157

A.17 The next data capture phase saw the addition of side-facing cameras and the mounting on a vehicle to reduce the effect rough surfaces. The cameras were also upgraded to global shutters to eliminate integration artifacts due to rolling shutter. 158

A.18 Final version used in this work consists of a fully self-contained weather resistant logging platform capable of deployment on any vehicle. Image capture is from a forward facing stereo pair and two side facing mono cameras. Camera synchronisation is controlled via a 5v fixed pulse rate signal from an Arduino microcontroller. Final version mounted on vehicle can be seen in Figure 4.2. 158

List of Tables

4.1 Mean post-BA reprojection error (px) of all features over variable length sequences using three different strategies for input point filtering prior to bundle adjustment. 69

6.1 Results showing comparison between different methods in terms of reconstruction rate. The S and D in parenthesis denotes which version of our approach is being used, S referring to the sparse feature point based reconstruction, D is with the addition of the dense stereo-from-motion phase. Time is processing time in seconds, t/c is in seconds per camera and n/t is reconstructed 3D points per second. *Note: results set 2 used a nearest-neighbor radius of 40px and results set 9 used nearest-neighbor radius of 20px but used a faster Intel i7-6700K CPU.* 123

Abbreviations

Stereo Visual Odometry	SVO
Semi-Global Block Matching	SGBM
Block Matching	BM
Efficient Large Scale Stereo	ELAS
Structure from Motion	SfM
Simultaneous Localisation And Mapping	SLAM
Monocular Visual Odometry	VO
Optical Flow	OF
Karlsruhe Institute of Technology and Toyota Technological Institute	KITTI
Sum of Absolute Differences	SAD
Field of View	FoV
Light Detection and Ranging	LIDAR
Radio Detection and Ranging	RADAR
Bundle Adjustment	BA
Global Navigation Satellite System	GNSS
Global Positioning System	GPS
Inertial Measurement Unit	IMU
Globalnaya Navigatsionnaya Sputnikovaya Sistema	GLONASS
Graphics Processing Unit	GPU
Central Processing Unit	CPU
Three Dimensional	3D
Two Dimensional	2D
Miles per Hour	mph

Declaration

The work in this report is based on research carried out at the Applied Maths and Computing Group, Cranfield University, England and the School of Engineering & Computer Sciences, Durham University, England. No part of this report has been submitted elsewhere for any other degree or qualification and is all my own work unless referenced to the contrary in the text.

Acknowledgements

I greatly appreciate the support and the precious time my supervisor, Dr. Toby Breckon, has provided throughout this work in both the theoretical understanding and practical execution when collecting data. Not forgetting his perseverance with regular video calls and their own unique technical challenges to enable my remote working.

During the work I had the pleasure to meet some brilliant people. Many thanks must go to Dr. Pedro Cavestany for providing much guidance on the theory of Structure-from-Motion, a good basis on which I could build. I would like to thank Dr. Ioannis Katramados for feedback and encouragement, and providing early opportunity to put theory into practice with flexible work at COSMONiO Ltd. Finally, Dereck Webster for his assistance with early data logging prototypes and providing stimulating discussions on many science and technology subjects.

I would like to acknowledge the industrial sponsor ZF TRW Conekt for their financial support and flexibility with balancing work and PhD commitments.

Lastly, I would like to express my appreciation for my family and their support, specifically my remarkably patient other-half, Stephanie. Whether it has been the kitchen table littered with cables, cameras and computer parts or skipping trips to see other family members, she has not given it a second thought.

Chapter 1

Introduction

This thesis addresses the current research challenges relating to low cost 3D mapping of outdoor environments within the automotive sector for future autonomous road vehicles.

Selecting an appropriate dense stereo approach that performs well on real world data in an automotive setting is a key issue. We use a novel dense stereo evaluation method that leverages annotated objects (typical of those found in automotive environments) to create an error metric for several dense stereo approaches (Chapter 3). In order to keep hardware costs and installation footprint low we use a minimal number of cameras while maintaining resolution. Therefore we propose an incremental Structure-from-Motion process that uses optical flow from a monocular camera to create uniform sparse point clouds, and an additional online densification processing stage resulting in 3D point generation rates that exceed state of the art approaches (Chapter 4). Creating clutter free 3D scene maps that do not contain moving objects such as cars and pedestrians is an important factor in the output. Extending the results from our stereo evaluation work we create a novel technique for dynamic object removal that uses data already acquired for 3D mapping, therefore avoiding the need for expensive recognition or segmentation algorithms (Chapter 5). Lastly, we perform registration of non-overlapping cameras, using pose-graph alignment, into a common coordinate system combining all point clouds into a scale consistent global 3D model (Chapter 6).

1.1 Motivation

It is clear that there is an ever increasing demand for better, more accurate and up to date 3D mapping data. At the same time there is a global trend towards automation of vehicles and robotic platforms in general. While both highly accurate 3D data and autonomous robots exist, the technology has not yet been developed to a level to be democratised. Current large scale 3D scanning remains in the hands of experts with specialised capture and processing hardware. Autonomous vehicles also utilise highly specialised hardware usually in the form of LIDAR (Light Detection and Ranging), RADAR (Radio Detection and Ranging), and advanced deep learning frameworks [8]. Many automotive manufactures are rapidly developing, producing and selling the sub-components required for full autonomy, such as lane keep assist, auto braking, adaptive cruise control and pedestrian detection [9–11]. Current autonomous systems require a plethora of sensors such as accelerometers, gyroscopes, wheel rotation encoders, ultrasonic sensors, GPS, RADAR, laser scanners, and camera systems all fused into one, or more, powerful processing units [12–16]. An essential stage to their autonomous operation is the that vehicle must know the location of itself and the 3D nature of the scene around it. All of the listed sensors can be categorised by being used for mapping or localisation, with laser scanners and cameras straddling the categories. While rotating laser scanners are the preferred technology in terms of accuracy and coverage, they are hampered by their price, size, shape and placement requirements, often needing to be placed high above the middle of the roof to ensure good coverage all round the vehicle; impacting the aerodynamics and degrading arguably a vehicle's key commercial attribute - aesthetics. It is therefore sensible to pursue the camera as a means to performing mapping and localisation; after all, that is currently the best method being used to pilot any vehicle, human eyesight. The prevalence of automotive grade cameras means they are a mature product already certified for use on vehicles, available at low unit cost and easily integrated into existing vehicle systems, making passive automotive cameras an abundant source of large amounts of data. The two properties of reconstruction speed and density of an implementation are traditionally inversely related, with more data comes slower processing, where faster processing is achieved through

data reduction. The question of what a system should focus on comes down to the intended audience. Is the output going to be consumed by a human or an algorithm? The main difference is our ability to naturally perceive 3D data and ignore noise. An algorithm may require 3D data in the form of a mesh, colour data or a point cloud with a minimum/maximum density. As we will not be tackling the problem of interpreting the 3D data, we aim to produce point cloud data of sufficient density that can be easily viewed by a human. Forward facing dense stereo has good frontal coverage, however it obviously lacks the ability to reconstruct any other directions. Spherical optics allow imagers to see 360° around the vehicle; one limitation is they have a lower angular resolution compared to using a standard lens, for example a 45° field of view, on the same digital imaging chip. Another limiting factor of spherical optics is they need to be located in specific positions to observe the world around them without encountering large amounts of obscuration. Rotating laser scanners face the same issue, as well as the size, weight, power, and cost considerations.

To summarise, we aim to address the problems of dense reconstruction around a moving vehicle whilst balancing angular resolution with scene coverage, which are inversely related using the same sensor hardware i.e. for a given sensor a narrow field of view has high angular resolution and the same sensor with a wide field of view has a lower angular resolution. We also aim to do this with visual cameras that can address most size, weight, power, and cost constraints associated with the widespread uptake within the automotive sector.

1.2 Related efforts

There is a significant amount of work in the area of 3D mapping from optical cameras both from the academic and commercial world. From the seminal work of [17] there was little interest until processing power and camera technology were both mature enough to handle the processing load and provide the input quality to enable practical use of this technique. It was in the 1990's that we saw interest in 3D reconstruction start to grow again [18,19] and by the 2000's some significant advances [20–26]. Today, online 3D map viewing services (Figure 1.1) provide impressive 3D

models suitable for human viewing, however they are currently low resolution at the vehicle scale (i.e. curbs, low walls, bollards, and lamp posts are poorly mapped, all of which are significant hazards for a moving vehicle) therefore are of little use for street level viewing of the 3D data and even less useful for navigation of autonomous systems.

Other commercial grade photogrammetry solutions [2,27] offer the ability to create very high quality 3D models (Figure 2.4) that are typically used for anything from mining site surveys to digital archiving of ancient artifacts, however these are offline processes only. They are able to achieve high resolution mapping at the cost of scale, often just covering the region of interest rather than city scale reconstructions. Commercial approaches are usually built upon well established techniques with some custom proprietary optimisations. The ever increasing accessibility of high performance CPUs and GPUs enable less efficient commercial implementations to still be effective. From an academic standpoint we have seen improvements in speed creating real-time mapping solutions [28, 29] and significantly slower but very high quality 3D offline reconstructions [30, 31]. With no system being able to perform dense large-scale reconstruction of outdoor environments without taking days to process on large clusters [32] or banks of GPUs [30, 31], it is still an open problem to be solved. One could argue that it is just a matter of waiting for processing hardware to catch up, however the consumer demand is there now. The mobile platforms that these algorithms will be deployed on have strict size, weight and power limitations that prevent installing the banks of GPUs that are currently needed. This presents a number of challenges with further research on different approaches to 3D visual mapping are outlined in Section 2.1.

1.3 Summary

There is a clear trend towards autonomous vehicles and their need for up to date accurate 3D models of the environment around the vehicle to aid in their navigation. While there exists large scale 3D mapping solutions, the source images are usually airborne or satellite based, therefore they lack the reconstruction resolution

Figure 1.1: Google Earth [1] image showing 3D reconstruction of suburban housing in Milton Keynes, UK.

at the ground level. Current ground based on-vehicle mapping typically utilise expensive large LIDAR scanners and/or many cameras arrays to achieve dense 3D point clouds. The hardware required for these systems is large, expensive and far from aerodynamic or aesthetically pleasing. This work will aim to create dense point clouds or 3D models around a vehicle as it transits a scene, using as few cameras as possible that would be capable of being installed with a small footprint on a vehicle, therefore minimising any impact on aerodynamics and aesthetics. We utilise a range of image processing techniques such as dense stereo vision, Visual Odometry (VO), and Structure-from-Motion (SfM) (all detailed in later Chapters). During the work we use novel evaluation methods to select appropriate dense stereo algorithms and address the issues of removing dynamic objects from the 3D data, with a unique approach that reuses data already required for the mapping process. We create an SfM approach that rivals state of the art approaches for quality and exceeds their data generations rates. Finally, we combine dense stereo, SVO and our SfM approach to create scale consistent global 3D scene maps that are automatically aligned using pose graph alignment.

1.4 Thesis Contributions

The contributions in this work advance state of the art in the following areas.

- Evaluating the plethora of dense stereo processing techniques [33–35] and their effectiveness on real-world data [36] by creating a novel object-wise stereo assessment method (Chapter 3).
- Creating dense 3D reconstructions using Structure-from-Motion [31,37,38] and extending it to an online pipeline that exceeds 3D point data-rates of state of the art approaches (Chapter 4).
- Removal of dynamic objects from the dense stereo mapping process [39–42] by creating a generalized approach using minimal extra processing (Chapter 5).
- Creating world scaled 3D reconstructions, from low-cost cameras, around a moving vehicle by registration of pose graphs from synchronised non-overlapping cameras (Chapter 6).

1.4.1 Publications

Results from this work were published as the following:

- A Foreground Object based Quantitative Assessment of Dense Stereo Approaches for use in Automotive Environments (O.K. Hamilton, T.P. Breckon, X. Bai, S. Kamata), In Proc. International Conference on Image Processing, IEEE, pp. 418-422, 2013. (Chapter 3).
- Quantitative Evaluation of Stereo Visual Odometry for Autonomous Vessel Localisation in Inland Waterway Sensing Applications (T. Kriechbaumer, K. Blackburn, T.P. Breckon, O. Hamilton, M. Riva-Casado), In Sensors, MDPI, Volume 15, No. 12, pp. 31869-31887, 2015.
- Generalized Dynamic Object Removal for Dense Stereo Vision Based Scene Mapping using Synthesised Optical Flow (O.K. Hamilton, T.P. Breckon), In Proc. International Conference on Image Processing, IEEE, pp. 3439-3443, 2016. (Chapter 5).

1.5 Thesis Structure

The structure of the thesis is split into chapters that focuses on the separate techniques used in this work and their individual challenges.

Chapter 2 will review the existing literature on general 3D reconstruction techniques, dense stereo imaging, and Structure-from-Motion. Owing to the number of dense stereo methods, this chapter will not explore the full extent of each technique used, rather it shall outline the research field of dense stereo and provide an overview of the processing commonality to the approaches. Further discussion into a selection of dense stereo approaches and their respective benefits is carried out in Chapter 3, where we provide quantitative analysis of their accuracy and justify why a given stereo approach was selected for this reconstruction task. In Chapter 4 we outline our method of Structure-from-Motion and online densification in order to perform a vital stage to creating dense 3D reconstruction from side-facing monocular cameras. Further processing that tackles the problem of removing multiple instances of dynamic objects from the final 3D reconstructions is demonstrated in Chapter 5. Combining all previous stages allows us to perform an automatic alignment process and reconstruct the 3D scene around the moving vehicle from non-overlapping cameras as illustrated in Chapter 6. Finally, Chapter 7 will summarise and discuss the results, implications, and further work.

Chapter 2

Literature Review

Reconstructing the world around us from video has been a rapidly evolving area of research for the last decade [20, 23, 43]. Multiple approaches have been developed with the aim of helping machines create 3D maps or models of the world around them so that they may navigate semi or fully autonomously. Mapping technologies generally fall into two categories: active and passive mapping.

Active systems emit some form of electromagnetic (EM) radiation, typically infrared (IR) light or high frequency radio-frequency (RF) in the form of RADAR. These systems are split again into several sub-categories some of which include structured-light, Time-of-Flight, and phase detection. Structured light systems, such as consumer depth cameras [44, 45], typically use a laser to project a known, static, pattern into the scene being scanned, then detect and measure the amount the pattern is distorted. The amount of distortion can be used to infer the structure that caused the perturbations in the projection [44]. Direct Time-of-Flight (ToF) systems (LIDAR) use high-speed electronics to time the delay between emitting a pulse of radiation to when it is received again [46]. A slightly different approach is Range-Gating, whereby the camera sensor integration period is synchronised with pulse emissions so that the intensity received is related to the distance and therefore the time the pulse was reflected [47]. Phase detection methods use a coded modulation of a transmitted pulse and detect the return signal modulation to determine the time since reflection and therefore calculate the distance. Some systems use combinations of multiple approaches, RADAR for example uses range-gating, phase detection and

Doppler-shift for determining range and range-rate [48] (the term range-rate is used as Doppler-shift only encodes information about the velocity component parallel to the transmitted/received beam, to determine any other component of the targets velocity a tracking solution is required). Active systems can provide high resolution depth data at high frame rates but with the disadvantage of often suffering from interference, most notably from other active emitters or more generally any source that is of high enough intensity to overwhelm the emitted signal. In the application to 3D mapping from a vehicle, the most problematic source is sunlight and its high intensity IR component outshines the IR signal from active cameras [47]. The pulsed-encoded signals of RADAR and monochromaticity (single wavelength) of LIDAR make them less susceptible to interference from solar radiation, however RADAR has very poor spatial resolution and LIDAR systems are prohibitively expensive and large for widespread commercial use in the emerging consumer autonomy sector.

Passive systems do not use any artificial emissions or coded EM pulses to measure the scene, instead they rely on parallax, i.e. observing a given scene from two slightly different viewpoints. This can be achieved through either multiple cameras sampled at the same time or a single camera at different positions. These two approaches are the foundations used in this research. Prior work on these two methods are highlighted in following sections.

2.1 Stereo Vision

Stereo imaging, stereo vision, dense stereo imaging or stereo correspondence algorithms are all commonly used terms for the process of using forward-facing parallel cameras to image a scene and reconstruct it in 3D space. This area of research has received vast amounts of attention over the years with the number of algorithms now in the hundreds [33, 34]. To enable systematic comparison of different algorithms, common datasets were released with ground truth 3D data for benchmarking [22, 33, 34, 49].

A popular dataset known as the Middlebury dataset and evaluation suite [22, 49] shows the rapid increase in number of submitted algorithms over time (Figure 2.2).

With over 150 algorithms listed in version 2 alone it is a daunting task to select suitable dense stereo methods for the reconstruction tasks in this work.

Work by [36] highlights an important difference between the carefully controlled scenes of the Middlebury dataset and the application of dense stereo to real-world data in the automotive setting. Middlebury images are typically well exposed, low noise and high contrast images that contain lots of structure, making them ideal for stereo matching. The real-world data from [36] shows the challenging nature of outdoor photography with variable lighting, optical aberrations from the vehicle windscreen and vehicle motion, reflections and the type of scene being viewed; most notably the low contrast homogeneous appearance of road surfaces (tarmac, concrete) and grass. The automotive based images of [36] were processed using a selection of dense stereo algorithms to perform a qualitative study. From the results, it is clear that there are many factors that affect the quality of the stereo results and that indoor static benchmark images may not be a good indicator of performance on dynamic outdoor scenes. From this work we are able to select a few candidate stereo algorithms [3, 5, 50–54] to use for further analysis and determine which to use in our 3D mapping approach.

Around the same time as this work started, a new dataset and online evaluation system was published [34] that focused on automotive scenarios. Their quantitative analysis was performed on a large custom dataset they collected using colour and greyscale stereo cameras, GPS, IMU, and a 64 beam 360° Velodyne laser scanner. Stereo accuracy metrics are generated by utilising the laser scanner data and calibrated camera information (pose and intrinsics) in order to project the 3D data from the laser scanner into the viewpoint of the left stereo camera (Figure 2.1). The ground truth 3D points, then converted to a ground truth disparity or depth maps, are used to extract metrics about the stereo correspondence algorithm under test over the whole image. Limitations on the laser scanners elevation coverage and its placement results in only approximately the lower 60% of the stereo images is covered by ground truth data. One could argue that this is, by far, the region of most interest as it contains the surface in front of the vehicle and any ground based object. However, as Figure 2.1 shows, the objects of interest in this example are vehicles

and in many cases are poorly covered by valid data points due to highly reflective surfaces such as glossy paint and windows. The elevation resolution of the laser scanner is limited to 64 beams, the physical beam width, according to the manufacture's manual, is stated as "*well defined rectangular shaped spot that is approximately 4" wide by 2" tall at 100' distance*" or approximately 10cm wide by 5cm tall at a distance of 30.48m. At a scan rate of 10Hz each laser creates 2084 points over the 360° scan range, therefore at a range of 30.5m, the approximately 191.6m circumference is divided into 9.2cm wide measurement points, the approximate width of the laser beam. A typical camera and optical configurations could be approximately 60° field of view at a resolution of 1280 pixels wide supplying an azimuthal resolution of just over 21 measurements/degree. The laser scanner used in [34] achieves an azimuthal resolution of just 5.8 measurements/degree. From these approximate figures, for azimuthal resolution, it is clear to see that even modest digital imagers far exceed the angular resolution of expensive laser scanners. Other technical benefits of digital imagers over laser scanners are centered around the mechanical properties such as size, weight and power (often referred to as SWaP), each being significantly lower for a camera system, along with the lack of moving parts, provide much more realistic solution for a robust sensor. Other robust automotive grade sensors, already used in mass production, such as ultrasonic and RADAR sensors meet the SWaP requirements but are, again, significantly lacking in angular resolution. The final argument is cost, thanks to the unrelenting thirst for cameras in everyday consumer electronics, the production cost per unit has plummeted while the sensor pixel count and quality continues to climb.

Considering all those points, one can see our motivation for exploring the use of passive imagers for the source of our data in the reconstruction task.

Given the hundreds of different implementations of dense stereo correspondence, delving into the inner workings of each of the stereo algorithms is far beyond the scope of this work (the reader is referred to the work of [33–35] for a comprehensive lists of dense stereo algorithms). In this work we are mainly concerned about the performance of a select stereo algorithm and its ability to integrate it into a larger processing chain to achieve 3D reconstruction around a moving vehicle rather

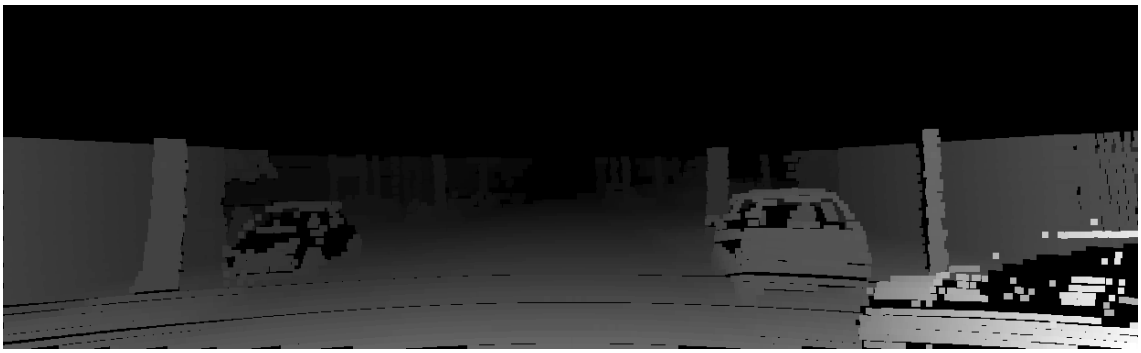

Figure 2.1: Example of KITTI laser scanner data projected into the left greyscale-camera coordinate system and scaled to simulate a disparity map.

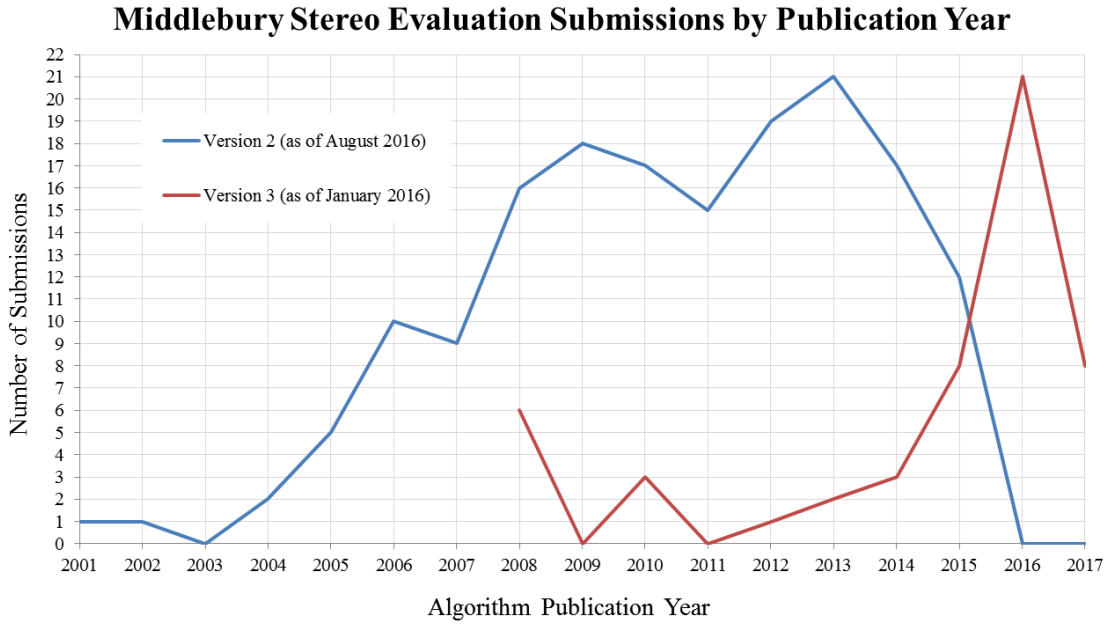

Figure 2.2: Number of results submitted to Middlebury evaluation (version 2 and 3) by publication year *(Note: May contain duplicate publications as some are submitted twice with different parameters).*

than designing another stereo approach that only provides marginal improvements. Chapter 3 outlines the dense stereo process of the selected algorithms.

2.2 Visual Odometry

Odometry is the general term for using motion data to estimate change in position of a platform over time to yield current position from the origin [25, 55, 56]. Typically data is obtained from contact based sensors and physically measures aspects of the platform such as rotary wheel encoders to measure the amount of wheel rotation [25,57]. Knowing the wheel radius and the rotation of each wheel, an estimate of the position can be calculated. Other odometry methods may use gyroscopes, magnetometers or accelerometers to determine the platform dynamics; however, these are not perfect. Wheel encoders are particularly sensitive to wheel slippage, this is where the friction between the wheel and surface in contact has been broken in some way such that the wheel may under-rotate or over-rotate causing an incorrect rotation value for the given position of the platform. A good example is a vehicle skidding on ice; the wheels can be stationary yet the car is continuing to change position. Solid state sensors (gyroscopes, magnetometers, accelerometers) are subject to external factors, like magnetic fields or vibration, and internal factors, like electronic noise, affecting measurement accuracy. Due to the nature of cumulative position updates, the current methods of odometry are prone to drifting away from the true platform position [14,25,57]. This drift is highly dependent on sensor quality, calibration and mounting of sensors and is typically quoted as a percentage i.e. a drift of 1% would mean a positional error of 1m after a 100m journey. Fusing data from multiple sensors can improve the accuracy [14]. Another common approach, that technically is not an odometry solution but yields the same results, is radio frequency based localisation most commonly used is GPS (Global Positioning System). However, commercial grade GPS locators are typically accurate to a few metres [58] which is sufficient for localising a moving vehicle to provide directions. For 3D mapping applications where the scale of objects within the scene is $<$1m, an odometry solution an with an error of an order of magnitude less is required. The

odometry approach we use is an image based one, typically called Visual Odometry (VO). There are two categories for visual odometry, monocular (usually just referred to as VO) and Stereo Visual Odometry (SVO).

2.2.1 Review of Existing Approaches

Visual Odometry is an image based solution for estimating the path of a camera. VO in general is a form of Structure-from-Motion (SfM discussed in Chapter 4) but is less concerned with the structure of the scene. It is robust to wheel slippage and has the ability to recover from drift in some circumstances. In this work we are concerned about the absolute scale of the reconstructed scene, in order to obtain absolute odometry, in metres, of the platform we rely on Stereo Visual Odometry (SVO). The main difference between VO and SVO is the ability of SVO to obtain scaled position information. In both techniques the underlying approach is to track image features across multiple frames and estimate the Fundamental matrix, F, [23] between successive frames and in turn use this to estimate the camera motion between frames. Typically VO alone does not allow for scale extraction [25] i.e. images captured from a monocular camera moving through a small scale model village will appear the same as those captured by a monocular camera moving through a real full sized village. It is possible to use VO for scale extraction, however the scale of either an observed object must be known or the camera height above the ground must be known [59], therefore constraining the camera placement, field-of-view to see the ground, or the scene being observed with calibrated targets of a known dimension. SVO has less constraint on camera position or views, it does however clearly require a second camera. This second viewpoint adds the scale parameter constraint missing from general VO, that constraint being the baseline between the stereo cameras in metres.

In stereo visual odometry 2D image features are matched between stereo left-right pairs and also matched to subsequent stereo pairs taken at a different position. Features are mapped between stereo pairs using the F matrix and 3D positions are calculated in a similar fashion to that done in the dense stereo imaging stage, using their 2D image positions in each image, the camera baseline, and lens focal length.

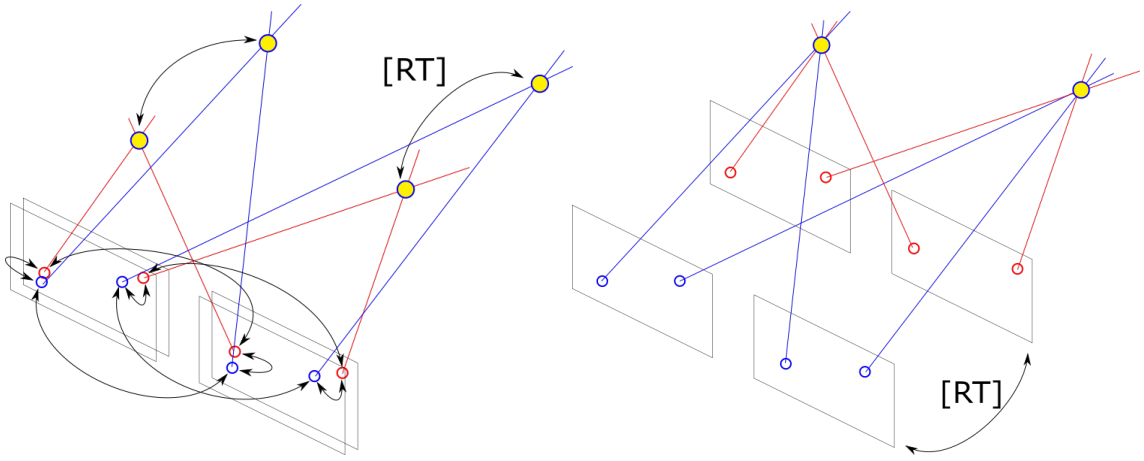

Figure 2.3: Illustration of feature tracking across stereo cameras in motion in order to extract scaled odometry.

The temporal feature matching to subsequent frames must triangulate to the same initial 3D locations, calculated from the previous frame, transformed by a rigid transformation (Figure 2.3). This rigid transform is the motion of the stereo camera rig. Figure 2.3 demonstrates the two parts to recovering the scaled odometry via SVO. In the left, we see feature pairs are matched between the left and right images (blue circles). The same two features are identified in a subsequent image pairs (red circles). In each stereo pair instance their 3D locations are triangulated (yellow circles). In the right, we show that a rigid transform (rotation and translation, [RT]) is calculated and applied that is required to align the common 3D points (yellow circles). The 3D transform required to align the 3D points (yellow) must also apply to the images and therefore the camera. The constant baseline between stereo images ensures the scale consistency of the 3D points.

The work of [25] introduces the term VO and presents a real-time solution for both monocular and stereo versions. They achieve an accuracy of between 1-5%, with respect to the odometry solution provided by a DPGS (differential global positioning system) [60] on three different outdoor datasets. They use simple but strong Harris corners [61] for feature point tracking. As images in video sequences are captured sequentially and from similar viewpoints (under relatively smooth motion and sufficient frame rates) Harris corners vary little from frame to frame, and therefore provide fast and stable points to match between. The same applies to Stereo Visual

Odometry (SVO) where the differing view from left to right is negligible enough that simple features are similar enough to be used. A further benefit of stereo based techniques is the row-aligned calibrated nature of the images [62]; this allows for the feature matching search space to be reduced to finding pairs that occupy a common row (or row range to allow for noise or calibration errors) in the left and right images. Further matching constraints can be applied to the column in which a matching feature can be located too, much like the disparity search limit imposed on the dense stereo. Both row and column constraints on possible feature locations allows for fast matching, and for simple point descriptions to be used as ambiguity between candidate matches is significantly reduced. This approach is seen in the later work of [63] where they use a simple 5×5 blob detection filter and a 5×5 corner detector. Matching features is done by comparing the sum of absolute differences (SAD) of vertical and horizontal Sobel responses in a block 11×11 centred on each feature. Further speed increases are found by restricting the SAD from comparing the full 121 pixels of the 11×11 window to just a sparse collection of 16 pixels within the 11×11 descriptor window. An epipolar error threshold of 1 pixel is used for initial candidate feature culling. They employ 'circular' matching as a confidence check whereby a match is initiated from the current left image and matched to the previous left image. From the previous left image it is matched to the previous right with the previous right being matched to the current right. Finally, the current right is matched to the current left image, hence the circular nature of the matching strategy. If the current left feature at the end of the matching circle is not the same feature as the initial one the match is declared invalid.

One limitation of image based positioning is the case that many of us have likely experienced ourselves as passengers in a vehicle. Looking out of a side window onto a scene where our FoV is mostly occupied by a large vehicle, such as a lorry or truck, can sometimes confuse our sense of motion when it begins to move. Depending on the relative velocities of our vehicle and the truck we experience different phantom motions. VO and SVO are both susceptible to false motion estimation if their view of the world is dominated by moving features [56,64,65]. The addition of accelerometers would be one option to detect if the camera is in a non-inertial reference frame

(a reference frame that is undergoing acceleration). Omnidirectional cameras can still provide ego motion under these conditions as their field of view increases the likelihood of observing static scene components [66].

In this study we do not specifically tackle the problem of stereo odometry as an isolated task, we instead utilise the existing approach of [63] that is capable of providing position updates with the speed and accuracy that is comparable to other positioning systems that we have at our disposal. As mentioned previously there are many technologies and approaches to position and odometry measurements, the most readily available and low cost of these being a GPS, however as we focus on a purely image driven system that is robust to problems of GNSS (e.g. cold start time, sky obscuration, and large positional error (>1m)) we use SVO for platform pose information.

2.3 Structure from Motion

As with stereo vision, the development of Structure-from-Motion (SfM) techniques have also grown rapidly over the last two decades since [18,67]. Where stereo image pairs are separated in space but synchronised in time, SfM images are separated in both space and time, hence the single camera must be under motion for both dimensions to change as a minimum requirement [23]. Stereo has the advantage of being able to reconstruct moving objects whereas reconstructing non-static scene components via SfM is a very difficult problem that has only recently seen some limited success [68]. On a hardware level, SfM is far simpler solution, only requiring a single moving camera. Cameras configured as a stereo pair must be calibrated and rigidly fixed in place, as any change in their relative rotations and translations has a profound effect on the performance of the dense stereo reconstruction, often causing complete failure [23]. They also require the images to be synchronised to a level appropriate for the application (e.g. for slow moving scenes or objects the synchronisation can be relaxed compared to a fast moving scene) but is still usually expected to be approximately <1ms time difference between frames. For example, at 70mph ($\sim$ 31m/s) a difference of 1ms corresponds to a distance of 3.1cm traversed, a

significant size compared to the baseline of the stereo cameras therefore invalidating the stereo results. As with stereo, SfM performs best with a global shutter camera. A camera in motion with a rolling shutter sensor will sample each row (assuming rolling row integration) at different positions in space, therefore using all the features in the image for solving camera poses results in failure as no single pose satisfies the epipolar geometry [23]. Under small or smooth camera motion it is possible to recover scene structure and camera poses using a rolling shutter camera [69, 70], however it is clear that SfM approaches built for global shutter cameras have higher performance than rolling shutter aware methods. Given the low maturity of rolling shutter aware SfM using global shutter cameras is the sensible option.

2.3.1 Review of Existing SfM Approaches

In general, the approaches used in different SFM methods are categorised as being in (but not limited to) the following groups:- Incremental, Global, Hierarchical, and Non-rigid. Each general approach has a subset of problems that breakdown into, graph optimisation, large scale matching, and unordered feature matching depending on their target application (i.e. large scale and unordered feature matching is less of a problem for sequential sequences). Regardless of the approach, the overarching common theme is built around the idea of solving 3D point positions and camera poses that satisfy the matched 2D feature locations in each image.

Global approaches to SfM are often thought to outperform incremental techniques [71–73] due to being less susceptible to drift and error as they solve the registration of all images simultaneously. Incremental methods, by their very nature, can suffer from accumulated drift as new images are added to the sequence [25]. In order to reduce the drift frequent Bundle Adjustment [20] passes must be executed. Bundle Adjustment (BA) is a large scale optimisation problem that solves multiple parameters such as camera poses and intrinsic properties with the goal to minimise the reprojection error of all the 3D features into 2D image space from which they were triangulated. Global approaches such as [71–79] are primarily concerned with producing a reconstruction from a collection of usually unsorted images. They are often closed problems such that there is a given number of input images that are

processed as part of a batch. The consequence of global processing is that intermediary data is often not available until the processing has been complete, this means that the pose graph information is not available during the SfM processing, meaning they are well suited to offline datasets. This global approach is satisfactory for city scale, unordered feature matching, however incremental methods can also process this type of dataset [26,37] as well as video sequences like the type one would expect from a moving camera rather than a collection of unordered images.

Hierarchical SfM, as with global SfM, requires a dataset is present before reconstruction processing can commence [80,81]. A tree structure is populated with images that are clustered by amount of overlap, therefore the likelihood of having many shared feature points [82]. The solving of camera poses and 3D points then commences from the leaves and progresses back up the tree. In order for this tree to be constructed all data must be present prior to the reconstruction phase. Again, like global SfM, this approach lends itself well to efficient matching of fixed size datasets.

Both global and hierarchical approaches have the challenge of unordered images as in input data and as a result are faced with the difficult task of unordered feature tracking [83,84]. This means any image has equal chance of matching to any other image. As discussed, hierarchical methods create a similarity tree structure to minimise this problem, however they face a similar problem when creating the tree. Strategies have been developed to tackle this problem [85], however this unordered matching problem will always be a problem when reconstructing an unconstrained image collection.

Non-rigid SfM is mainly concerned with reconstruction of non-rigid objects, a difficult task for other SfM methods as they all assume that the tracked feature points are static with respect to the moving camera in order to satisfy the epipolar geometry and enable camera motion estimation [23]. However, one of the challenges with automotive data is the fact that road environments are not clean static scenes, they contain other moving vehicles and pedestrians. In rural situations, where most of the visual landmarks are natural objects such as trees, bushes, and hedges, the wind or aerodynamic wake of the vehicle, can dramatically change the position of

the features being tracked. Work by [68] investigates non-rigid SfM, although it is not in an automotive setting and focuses on the reconstruction of small objects, it demonstrates a possible processing method that may be applicable in rural dynamic scenes. Performing SfM in the presence of highly cluttered scenes with dynamic deformable objects is beyond the scope of this work but is acknowledged as a significant problem that should be addressed in future work.

In our work we are reconstructing scenes in 3D from an unknown dataset size as the vehicle traverses a scene, therefore an incremental approach is used. Incremental SfM offers the ability to add more images into the processing chain at run-time, making it a candidate for online processing. The work of [26, 86] is designed to create a system where users can easily explore large image collection in a 3D manner. Instead of viewing a collection of photos chronologically the user can view them geographically. They take large publicly available images from famous landmarks and reconstruct the 3D scene and poses of each camera used. This is a significant task that is reflected in the processing time required to perform such a calculation, 2 weeks to successfully match and reconstruct 597 cameras from a dataset of 2635 and a few hours for a smaller set of 82 out of 120. There are two main factors that contribute to this run-time, the first being the unordered nature of the image set, without being reliant on global positioning information, such as GPS, to prime candidate feature matches there is a large number of feature descriptors that must be matched across all images in the dataset. The second, and biggest impact on run-time, is the bundle-adjustment (BA) process. As their input datasets are created from an ensemble of camera manufacturers each with different resolutions, focal lengths, and optical distortion characteristics, this adds many extra parameters that must be optimised within the BA phase. More recent work on large scale reconstruction from [38] attempts to increase the speed of reconstruction by leveraging parallel BA. They present results showing the processing of 15k images in 1.67hrs (using 24-cores of Intel Xenon at 3.33GHz and a GTX 480 GPU) and acknowledging that feature matching over thousands of images is now a significant time component within their processing. Another time expensive stage is the reorganisation of point lists to associate between 2D and 3D points for resectioning, where new cameras are

added into the processing and their newly matched 2D features must be added to the already populated 2D matching list and associated triangulated 3D points.

To evaluate performance they use number of cameras successfully solved per unit processing time and the number of 3D triangulated data points per unit processing time as metrics to compare implementations processing different datasets. As with any processing speed comparison it is extremely limited in usefulness as processing hardware varies greatly. To perform any reliable comparison a standardized hardware test-bed would need to be established across the field.

The latest work by [31] revisits the problem of large scale reconstruction using incremental methods. The additional stages they contribute are a next best view selection, where they select the next best view that would increase robustness and accuracy of the BA process. A naive best frame selection can be decided by simply picking the frame with the greatest number of features, however this could easily be skewed by a frame containing a particularly feature strong object, other factors such as potential triangulation accuracy and uniform distribution over the image are important factors. In order to preserve feature distribution uniformity an image region binning strategy is used where the image is split into $m \times n$ rectangular regions and the only strongest k features are kept for processing in the SfM pipeline.

A common theme throughout the prior research discussed here is the FoV of the cameras used are typical to standard cameras (i.e. they assume standard camera pin-hole models). There are a number of SfM approaches that use other camera models, most notably, fisheye lenses to provide FoVs typically of $\sim 180^{o}$. While fisheye cameras provide good opportunity for localisation thanks to wide spread feature points they suffer from greatly reduced angular resolution, this manifests are poorer accuracy at greater ranges. A good example is seen in [87] where they have successfully mapped around the vehicle with good scene coverage thanks to the fisheye optics, however, examining the quality of their point clouds shows the usable range only extending to the roadside, a few metres away from the cameras.

Much of the work discussed here and other sparse mapping [88] provides 3D models in sparse or semi-dense format, another process must be used [89] in order to obtain the dense point clouds or meshes, making these processes offline tasks.

Incremental SFM methods offer the best chance for dense online 3D mapping of an unknown sequence. By unknown we refer to the fact the sequence length is undefined prior to starting the SFM process, this allows incremental approaches to theoretically run indefinitely under the constraint of memory limitations.

An optional final stage of SfM processing is to perform loop-closing or pose graph optimisation [90,91]. This is generally more applicable to Simultaneous Localisation And Mapping (SLAM) [24,28,40,92] where an exploratory path is taken to visit as many areas as possible to build a complete map of the area that can then be used in parallel for localisation, a similar process to that one might expect, for example, in an autonomous vacuum cleaner [93]. The problem loop-closing solves is the accumulated drift of camera pose that arises from imperfect feature or flow matches and frame-to-frame pose estimation. As the name suggests, this stage requires that the camera revisits a part of the scene, therefore performing some form of a loop in the pose-graph. When a previous scene is recognised the pose-graph is globally optimised so that the current camera position is aligned to the previously recognised position. In general this can not be applied to VO as feature points and tracks are not kept beyond a few frames when which they are no longer required for matching to the latest frame from the camera, preventing historic matches being made and loops being closed.

While loop-closure has been demonstrated to be very effective at realigning large scale reconstructions [94,95] the application to real-world problems is very scenario dependent. For mapping tasks, loop-closing makes a significant difference, however it does require the route to be planned or weighted towards maximum coverage so that previous sections are revisited. For autonomous driving tasks there is very little opportunity to revisit previous points along the pose graph as a typical journey in a car is a point-to-point path with very few trips encountering the same scene twice in one journey. However it could provide useful for long term usage on well travelled routes, such as commuter journeys.

2.4 3D Reconstruction

The techniques described in the previous sections, dense stereo, VO, SVO, and SfM are all tools to be used in the general task of 3D reconstruction. The process as a whole is known as photogrammetry. The term is usually used in commercial products that incorporate one or more of the above elements to produce a mesh model from images or video. As briefly described in Chapter 1 there are multiple prior examples of high quality 3D reconstructions from commercially available software [2, 27] and public examples used everyday on products like Google Earth [1]. Commercial applications often require large amounts of processing power and time to output 3D models, both of which are minor issues for offline processing, hosting of the end results and presenting them as a map, such as the case with Google Earth [1]. In this section we examine prior work aimed at point cloud processing, surface computation, mesh construction and texturing. However, much of which only serves as a rendering method for human consumption of the reconstruction output. Figure 2.4 demonstrates the difference between 3D reconstruction stages and their intended consumer. The top images contain the raw point cloud data, i.e. all the 3D information about the scene. The top left shows the raw feature points used for the camera pose estimation in the SfM process, whilst there are a moderate number of matched features (919) the 3D rendering is very sparse as they only represent 0.07% of the pixels from a single frame. Top right is the dense point cloud that numbers 672,416 individual 3D points, much more of the structure can be seen when rendering and is more human friendly for viewing. Bottom left is a triangular mesh created using the dense point cloud, there are 20,000 triangles that are naively coloured from the dense point cloud. Finally the bottom right is the mesh fully UV mapped an textured using the input images. UV mapping is the process of calculating which parts of a texture should be displayed on each triangle of the mesh, whereas previously the triangle had only a single colour assigned to it. This results in the original input images from the camera being displayed over the mesh data providing high quality reconstructions ideal for human consumption. If the purpose of the reconstruction is only to determine the presence of objects or navigable areas then sparse processing may suffice or a tunable densification approach to achieve

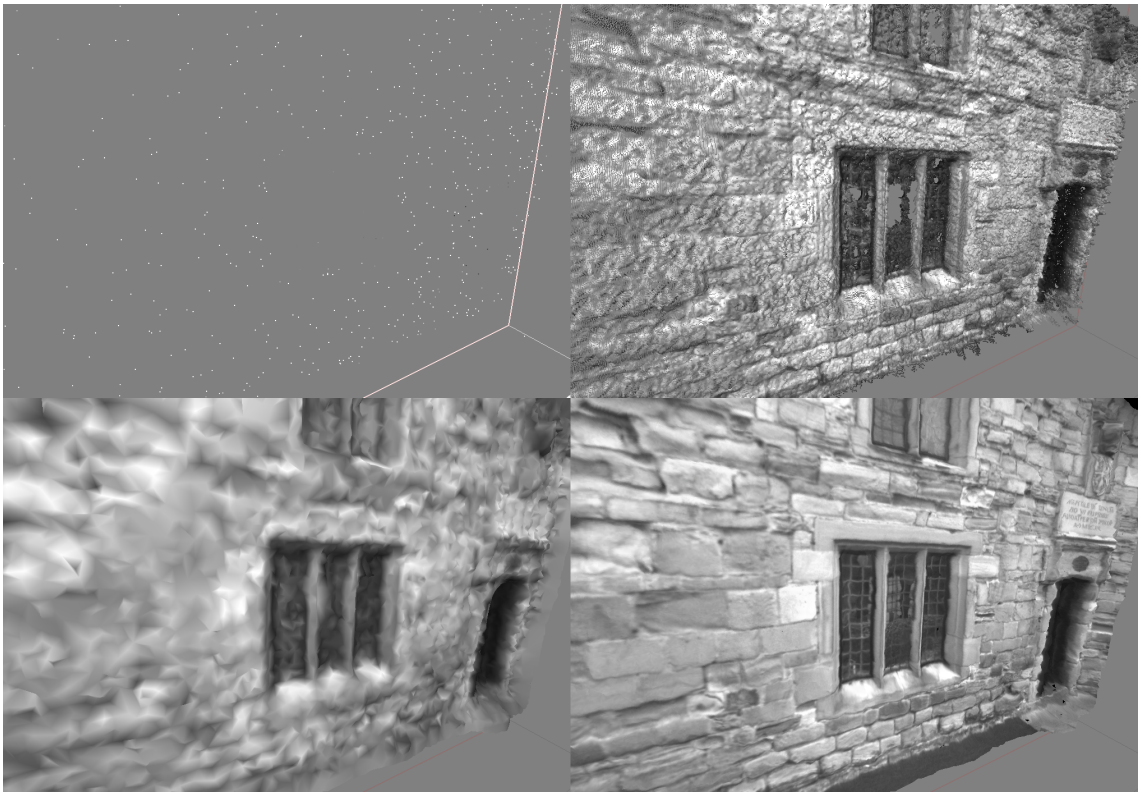

Figure 2.4: Photoscan [2] reconstruction using two input images. Top Left: Sparse point cloud from feature points used in the SfM process. Top Right: Dense point cloud from multi-view stereo process. Bottom Left: Mesh constructed from dense point cloud with mesh triangles naively coloured and lit. Bottom Right: Mesh with correctly UV mapped texture and calculated mesh normals to allow for correct lighting.

different levels of point cloud density as required.

2.4.1 Review of Existing 3D Reconstruction Approaches

As discussed previously there are many different sensor options to use as an input for a mapping solution. An illustration of the capability of laser scanners to perform 3D mapping is shown in [12]. The obvious limitation of laser scanning is the lack of colour or even greyscale information to build a textured mesh, it does however provide fast and accurate 3D measurements, ideal for consumption by a machine. The advantage of having raw data already in 3D form is it does not require the processor-intensive image-based algorithms, outlined previously, to generate the 3D points. This leaves more processing resources free to address the task of interpreting the 3D data. The work of [12] also demonstrates the typical size requirements of

LIDAR systems and their significant impact on aerodynamics and aesthetics.

Work by [32] shows promising results that are very close to achieving the end goal of 3D mapping from a transiting vehicle. They use 'camera quadruples', these are units of four cameras on a mounting plate with minimal overlapping Fields-of-View (FoV). One of the middle two cameras faces directly outwards from the plate, the second is elevated to provide more vertical coverage. The outer two cameras are angled in towards each other in a 'toe-in' configuration. Their data capture solution consists of two 'camera quadruples', one on each side of the vehicle, making their system consist of 8 side facing cameras. The quality of their 3D reconstruction appears to be very high post meshing and colouring. Their processing target time is stated as "*to process up to 6 hours of acquired data in an equal amount of time*" which by definition is real-time processing, however, they refer to 6 hours as being total video recorded across all cameras (i.e. 45 minutes from each camera or 45 minutes of driving time processed in 6 hours). Their largest example of 3D output model shows a path length of approximately 200m, assuming the processing time referred to this dataset that gives them an approximate driving speed of 0.27km/h. The hardware used to process the data is also listed as a 10-PC cluster of dual-core machines; despite this work being from 2006, that amount of processing power is still significant by today's standards. Another limitation with this system is the lack of forward facing cameras therefore failing to map the terrain in front of the vehicle. Commonly cited work is that of [89], often referred to as PMVS (Patch-based Multi-View Stereo), the name of the software created by the authors and powered by their algorithm in the referenced work. Taking, as an input, camera positions and images it generates a series of oriented patches for its multi-view matching strategy and generates very high quality models. An update to PMVS, CMVS (Clustering Views for Multi-View Stereo) [30] designed to reduce the memory footprint and improve speed by clustering into overlapping sets and process them parallel. While the results shown are impressive, noise filtering, a 3×3 up-sampling of the CMVS points, and QSpat [96] are all used to improve the rendering of the original point clouds. Run-times listed by CMVS exclude the SfM processing stage as they use a preprocessed data source that supplies camera poses, a sparse set of

3D points, and a visibility list indicating which points are visible in each camera. This approach is well suited to densification of offline global SfM approaches. An approach by [97] shows comparable results to [30], adding the ability to grow the density of reconstruction by allowing the process to run for longer. It also provides a means of prioritising regions for reconstruction resolution, a feature that may be useful to automotive data by increasing the priority of ground level reconstruction and decreasing the resolution of the scene above the roof-line of the vehicle. The recent work of [31] demonstrates the state of the art providing very high quality reconstruction on large scale outdoor scenes, however, again this is not a rapid process taking 4.2 days for the dense reconstruction process running on 4 Nvidia Titan X GPUs after an initial 6 days of processing for sparse SfM processing.

2.5 Dynamic Object Removal

Moving objects are easily identified in static videos using various forms of image differencing or background subtraction. Those techniques can still apply to cameras that experience small amounts of motion. Traditionally frame alignment would be performed by applying a 2D homography transform, essentially performing image stabilisation [98,99] allowing for frame differencing to be applied. The unconstrained motion of a camera moving in free space precludes the use of such transforms for performing stabilisation.

A consequence of various mapping approaches is that moving objects are seen multiple times and at different locations, this results in the object being reconstructed multiple times throughout the global map. Prior work attempts to address the detection problem in terms of identifying the location and approximate region occupied by a moving object [39–42]. This is effective for use in autonomous vehicles to aid with obstacle avoidance, however for use in mapping applications the segmented region often fails to completely cover only the moving object [42]. Structure-from-Motion (SfM) mapping methods broadly rely on feature tracking and estimation of extrinsic camera parameters to calculate the pose of the camera and thus enable triangulation of the tracked features. Typically when selecting features to track, a

RANSAC [100] process is used to filter for valid static background matches, where the background typically contains the majority of the image features. The SfM approach to mapping naturally rejects foreground dynamic objects at the RANSAC stage. In contrast to SfM, 3D structure recovery from dense stereo means both static and dynamic objects appear equally within the temporal alignment of the stereo pair. The work of [41, 42] both explore the task of moving object removal by using feature tracking to identify targets and depth samples to detect candidates. Tracked features in the image sequence with observed motion that does not conform to expected static scene motion, estimated from the cameras ego-motion, are declared as dynamic. Whilst this proves effective in detecting candidate regions, the sparse nature of the feature points mean further analysis of the original intensity imagery is required to correctly segment the object, thus increasing both complexity and processing load. Accurate segmentation is an intensive processing task and is heavily dependent on several factors such as texture, lighting variations, shadows, and reflections [42]. The work by [101] demonstrates the inherent use of stereo vision for moving object detection. However it is limited to use within a basic stereo processing technique for moving candidate confirmation. Whilst such block based approaches have been shown to perform comparatively to contemporary approaches [102], this dependency limits the wider applicability of [101] to a small subset of stereo algorithms in general [50]. Furthermore, the moving object mask produced from [101] is sparse in nature and is insufficient to effectively remove the object from a dense disparity map. Work by [103] demonstrates moving object masks that are more dense than [101], however still require a separate optical flow calculation in intensity space. Moreover, the frame rate of [103] remains unclear. Comparison of frames separated by larger time intervals (i.e. a lower frame rate) will aid in detection of slow moving objects or objects that move perpendicularly to the camera motion as the depth values across the object will not vary significantly frame-to-frame. The most notable recent work by [104] comes closest to achieving the dense segmentation required to perform the removal of moving object from the mapping pipeline. Their approach is a far more complex optimisation strategy that first formulates a *motion field model* that derives a relationship for 2D image plane

motions from 3D motions of scene objects, then estimates the possible 3D motion of the scene objects in a pixel-wise manner. However, the estimated 3D motions are non-unique and require further constraint to be consistent with physical, real-world, motion properties of the system. Constructing a series of energy terms, relating to the possible velocity of each pixel and enforcing a smoothness constraint, a minimisation is performed to obtain the velocity field of the image. A further step of clustering is carried out on the velocity components to create the final segmented regions corresponding to moving objects. Furthermore, this approach relies on creating a disparity map from the laser scanner data, therefore creating a disparity map with far lower noise and holes than one created with a traditional dense stereo algorithm and adding extra hardware complexity and cost.

2.6 Summary

From the early work of [18] geometric reconstruction from images has seen a vast amount of attention in all areas from stereo vision to SfM to point cloud meshing and rendering. With the exponential growth in processing power, increased resolution of digital imagers and the ever decreasing price of both technologies lay the foundations for the widespread development of further complex processing and algorithms for 3D reconstruction. High resolution reconstructions still take on the order of days to process and online dense reconstructions are typically low resolution or constrained to limits on the reconstruction size. With highly capable hardware now readily available, over the course of this work there have been many significant developments within the field of 3D reconstruction from images. The approaches detailed in subsequent chapters are designed to address problems identified during this research. The following paragraphs highlight some of the specific issues identified in prior work that we aim to tackle in our approach.

Stereo Vision

As mentioned in Section 2.1, with the existing number of dense stereo approaches there is little benefit to our final reconstruction output by creating yet another approach. As a result we perform some evaluation on automotive data and identify

one or more suitable algorithms that will be used throughout this work (Chapter 3). Similarly as VO and SVO solutions have demonstrated positional accuracy on par or better than GNSS solutions we review existing implementations and select the most appropriate for our solution.

Structure from Motion

Along with stereo, the number of SfM approaches has grown rapidly, however it is still an open problem with no generic solution for large-scale, online, dense reconstruction. Dense real-time SLAM exist for small indoor scenes and outdoor environments have seen sparse SLAM solutions. Many high resolution dense outdoor approaches are built around global SfM approaches that are processed offline, taking from several hours to several days. The following list contains key issues identified from multiple prior work in Section 2.3 that we aim to tackle with our implementation of SfM (Chapter 4).

- An incremental SfM solution that uses small adaptive BA window sizes to avoid the traditional large windows associated with incremental approaches [26].
- An approach that allows for simple rapid reorganisation of new 3D-2D point correspondences as frames are added [38].
- Remove the need for region binning while preserving uniform feature point distribution [31].
- Perform online high resolution densification using dense stereo approaches that adds minimal extra processing time [30].

Dynamic Object Removal

Using dense stereo vision for mapping urban environments in the presence of moving, or dynamic, objects has the unwanted consequence of reconstructing the object at multiple locations. General recognition of objects is an open problem with a significant amount of prior work. We tackle the issue of object removal in a generic way that is independent of object class or motion. Prior work in this area outlined in Chapter 5 along with our novel approach.

Chapter 3

Stereo Vision for 3D Mapping

This chapter will explore the background and use of stereoscopic imaging techniques for 3D reconstruction, stereoscopic camera pose estimation and odometry in automotive applications, current approaches, and limitations. We also present a novel stereo evaluation technique for dense stereo algorithms using foreground objects.

3.1 Dense Stereo Imaging

Dense stereoscopic imaging, often just referred to as stereo imaging, dense stereo or similar, comprises the use of two (or more) cameras separated along an axis perpendicular to the view direction, effectively mimicking the configuration of binocular vision found in nature such as that used by humans. In this work all stereo imaging is performed using horizontally separated cameras and referred to as stereo left or right as viewed from the behind the cameras. In general, dense stereo processing is performed by matching pixels of the left image with their corresponding pixels in the right image. The relative positions, along the x-axis, of matching image components is used (along with the baseline separation of the stereo cameras and their focal length) to triangulate a position in 3D space (Equation 3.1). This process is performed over all pixels, imaged by both left and right cameras, to create a dense depth image or 3D point cloud information.

There has been significant interest in dense stereo correspondence algorithms in recent years for use in the urban automotive environment [34, 36, 105]. As of July

2012 the online comparison service "The KITTI Vision Benchmark Suite" [34] listed 10 different dense stereo correspondence algorithms, as of February 2016 it boasts results from 96 different implementations of dense stereo imaging. One of the attractions of dense stereo is the ability to extract real-world scaled scene information from low-cost high-volume consumer grade devices. Previously, extracting such information would require an expensive and bulky scanning laser device at the same time sacrificing image information that can be leveraged by other processing strategies such as machine learning techniques for speed sign, road marking and traffic light recognition, to name but a few.

In the following sections we will discuss in further detail the method behind all stereo vision for 3D mapping, the current state of the art, their performance and known limitations. There is an ever growing pool of dense stereo algorithms, as illustrated by the increasing numbers listed on [34] and Figure 2.2, of which most can be summarised by four processing steps: pixel matching, matching cost aggregation, disparity measurement and post processing. Examining these processing stages of all the different dense stereo algorithms is beyond the scope of this work, as such only a high level description of dense stereo is discussed.

3.1.1 Matching Strategies

A prerequisite to performing dense stereo is that the images are undistorted and rectified. This is the process of removing barrel and pincushion distortion and row-aligning the images (for horizontal stereo) to reduce the pixel-matching search space from two dimensions, x and y, to only a one-dimensional problem searching in x. The most basic form of matching is to create a support window around the pixel under test, such a method is used by [50]. A test patch is selected from the left image at a given row and compared to the right image. This is where the rectification, or row alignment, of the images is important as misalignment would result in incorrect regions being compared. The test patch is slid along a given row (y) at different column positions (x) in the right image until a good match is found. Region matching costs are computed in various ways, typical approaches include Sum of Squared Differences or Sum of Absolute Differences (SAD). Support regions

can be generated in naive approach with a simple square window [5,50] or a dynamic patch can be generated using neighboring pixels of similar intensity or colour [51–53]. Along with image row alignment, another performance strategy is to limit the patch comparison to a range of x positions in the right image. Imposing a maximal allowed disparity search range has the effect of creating a minimal distance over which the algorithm can perform. For example, limiting the x-axis search range, in the right image, to just 16 pixels would only permit matching to distant parts of the observed scene, however this would have the benefit of shorter processing times. Setting the maximal disparity search range to 256 pixels, for example, would facilitate matching to much closer parts of the scene at the cost of far longer processing times and the potential for mismatches. Other factors that greatly impact performance, in both terms of processing time and quality of results, is the matching support window size. If the window is too large then fine scene detail is lost and processing time increases, however noise and mismatches are reduced. Conversely, a very small window size will extract fine scene structures at shorter processing times but at the cost of introducing noise from false matches.

3.1.2 Matching Cost

The matching cost as a function of image position are computed. As patch positions are quantised to integer pixel locations so too is the disparity map and therefore the 3D triangulated range measurements. To mitigate this quantisation the interpolation of matching costs are performed around the region of lowest matching score to provide a subpixel level accuracy for matching patch locations [5, 50]. Stereo correspondence algorithms generally fall into one of two groups, local or global. Local algorithms use a local support window to compare pixel regions, selecting a locally minimal matching cost [50]. By contrast, global approaches rely on energy minimization techniques to compute disparities, such as graph cuts [22,106] and dynamic programming [22, 53, 106, 107]. These approaches to dense stereo imaging are by no means exhaustive; very recent developments in Convolution Neural Networks, a form of machine learning, have been applied to the problem of stereo correspondence and have yielded some impressive results [108, 109] being highly ranked by [34].

3.1.3 Disparity Measurement

Regardless of the underlying technique, the output invariably takes the form of a disparity map. This map is a 2D matrix with index values relating to the x-axis offset, in pixels, of corresponding matches from the left stereo image to the right stereo image. Typically the range of values in the disparity map can be around 0-64, depending on the baseline of the cameras or an imposed maximal search space by the algorithm being used. Normalising a disparity map and scaling between 0-255 produces a greyscale depth map for human viewing, Figure 3.2. For parallel cameras, objects viewed at infinity appear in each image (left and right) at the same location; this case would equate to a disparity value of 0 or black. As objects approach the stereo camera, their relative positions in the left and right images will diverge, resulting in increasing disparity values which are normalised to 255 or white for viewing purposes only. The original, un-normalised, disparity map can then be used to create 3D points (Equation 3.1), using known parameters of the stereo camera configuration to project pixels into 3D scaled space.

3.1.4 Post Processing

Achieving a completely-dense measured disparity map is virtually impossible by the very definition of stereo vision. For stereo vision to work, a parallax view of the scene is required to enable triangulation; any practical real world 3D scene viewed from different angles will have foreground objects occluding parts of the background scene. In very specific circumstances, such as a smooth-walled corridor, there will be no occluders, however such a scenario is unrealistic in real-world environments. Some stereo implementations perform occlusion detection and estimate the values to fill in these empty regions. This additional processing stage can have minimal benefits if the source depth map is of a poor quality due to the complexity of the scene, or if the stereo camera is moving, providing different view aspects of the scene and enabling real measurements to fill in the missing data.

Figure 3.1: Sample of the input data (frame 17) from the left stereo camera in the KITTI dataset.

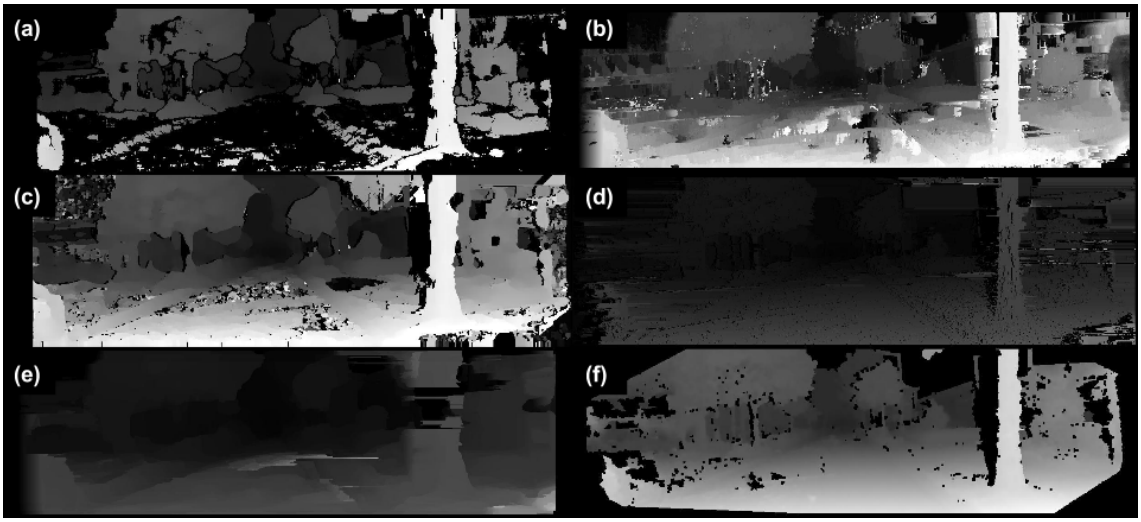

Figure 3.2: Example disparity maps of frame 17 from KITTI dataset from sequence 2011_09_26_drive_0009. (a) BM, (b) Cross, (c) SGBM, (d) AdaptDP, (e) NoMD, (f) ELAS.

3.2 Quantitative Assessment of Dense Stereo Vision

To manage the workload of the project we must reduce the number of stereo algorithms we use to only a select few. Prior to [34] the popular stereo benchmark data set was [49] which features a series of complex static indoor scenes. Work by [36] took a selection of algorithms and applied them to complex outdoor scenes in an automotive environment and compared the results in a qualitative fashion. The conclusion was clear that dynamic real-world scenes pose a far greater challenge to dense stereo processing and further research was required. Alongside the release

of KITTI stereo images came ground truth data in the form of 360° LIDAR [110] data and manually annotated object information in the form of 3D bounding boxes surrounding objects, e.g. cars, trucks, cyclists and pedestrians. KITTI performs stereo evaluation on a global image basis, comparing against a synthetic disparity image generated from LIDAR points. The limitation with this approach is interpolation between LIDAR points is required to generate a dense synthetic disparity map on which to perform the comparison; this inherently generates a large amount of estimated data.

We decided to provide a new benchmarking approach that examines the real world accuracy of dense stereo vision algorithms on foreground objects. We use a selection of stereo correspondence algorithms, (Block Matching (BM) [50], Semi-global block matching (SGBM) [5], No Maximum Disparity (NoMD) [51], Cross-based local approach (Cross) [52], Adaptive cost aggregation and dynamic programming (AdptDP) [53], Efficient Large-Scale Stereo (ELAS) [3] and Non-local Aggregation (NonLocal) [54]) and assess their reconstruction accuracy as a function of range. Previous comparative studies do not perform such quantitative analysis [34,36,105]. Leveraging the recent availability of annotated ground truth data [34], we proposed a registration based methodology, explicitly analysing the depth accuracy against range on foreground objects. Within the automotive application of stereo vision, such foreground objects (e.g. cars, trucks, cyclists and pedestrians) are of primary importance. Despite this, the evaluation on such objects is lacking in prior studies [34, 36, 105]. By contrast, we perform an object-wise quantitative assessment of dense stereo accuracy as a function of range, specifically targeting foreground object accuracy to offer new insight into relative algorithm performance.

Our quantitative comparison methodology has four stages based on: 1) disparity map generation from dense stereo technique of choice; 2) subsequent stereo point cloud generation via 3D triangulation; 3) foreground object segmentation; 4) registration between segmented stereo point cloud and ground truth, Figure 3.5 shows the process.

After the disparity map has been generated from a given stereo correspondence algorithm, each pixel is triangulated to real world coordinates, Equation 3.1, and

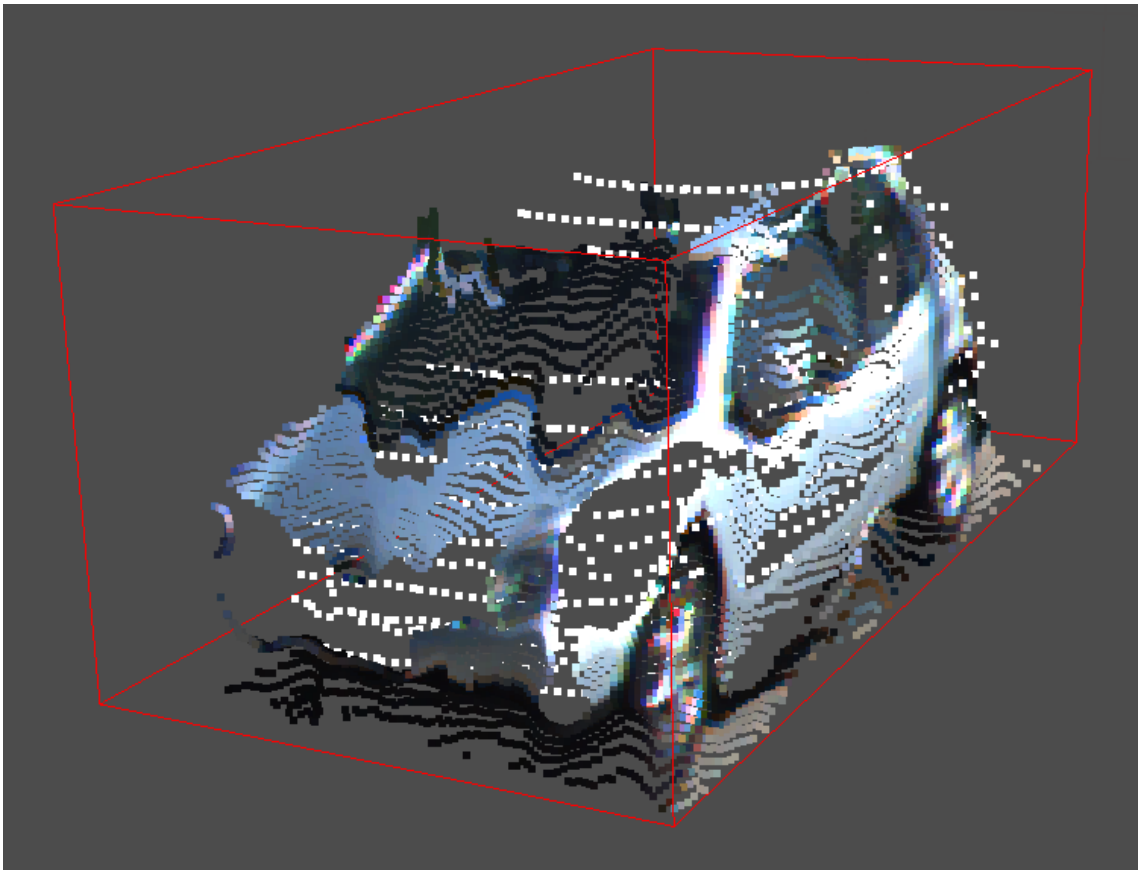

Figure 3.3: Visualisation of colour mapped disparity point cloud and laser ground truth point cloud (white points).

added to a disparity point cloud. The corresponding ground truth data, supplied from a laser scanner, is transformed into the left camera coordinate system of the stereo set up using the supplied calibration information [34]. This stereo disparity map to point cloud conversion is carried out using the following transformation:

$$X = \frac{Z(u - cu)}{f} \qquad Y = \frac{Z(v - cv)}{f} \qquad Z = \frac{fB}{d} \tag{3.1}$$

where f = focal length (pixels), B = baseline (mm), d = pixel disparity (pixels), $[u, v]$ = disparity map pixel x and y positions respectively (pixels), $[cu, cv]$ = image centre along the optical axis, $[X, Y, Z]$ = real world coordinates in camera reference frame (mm).

A hypothetically perfect dense stereo algorithm with error free measurements of disparity, d, will produce a triangulated 3D depth estimate matching exactly to point $P = (X, Y, Z)$ (i.e. identical to the ground truth). In practice, due to disparity

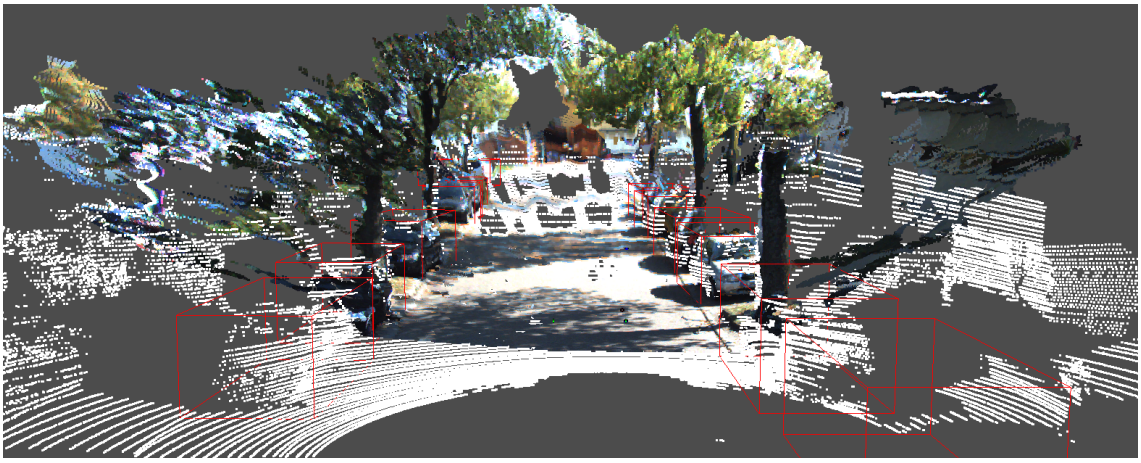

Figure 3.4: Colour disparity point cloud generated using [3] with laser scanner point cloud (white dots) and annotated ground truth bounding boxes (red boxes).

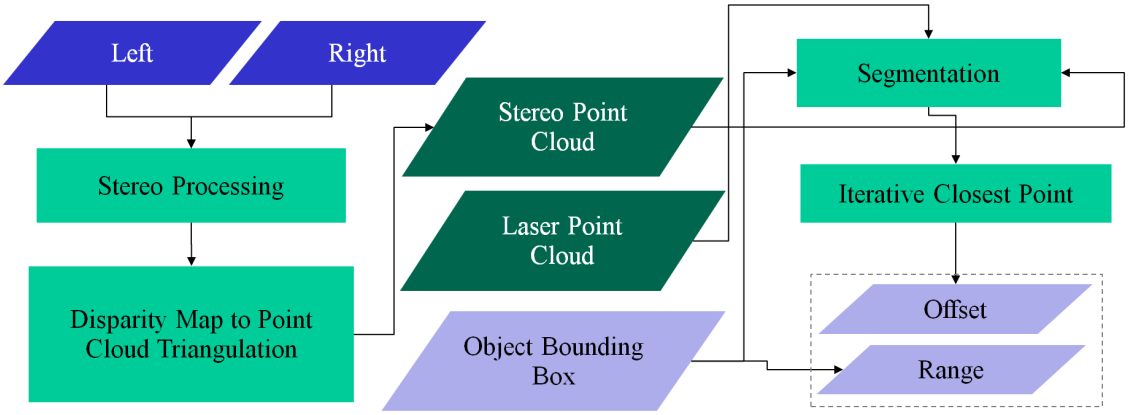

Figure 3.5: Flow chart for performing quantitative evaluation of dense stereo algorithms on foreground objects of interest.

errors in the stereo algorithms, this 3D point estimate is imperfect.

By differentiating Equation 3.1 w.r.t disparity, d, to recover the derivative of range against disparity we get the following:

$$\frac{\delta Z}{\delta d} = -fBd^{-2} \tag{3.2}$$

We rearrange Equation 3.1 and Equation 3.2 to form the following two relationships:

$$\Delta Z = -\frac{fB}{d^2}\Delta d \tag{3.3}$$

$$d^2 = \frac{f^2 B^2}{Z^2} \tag{3.4}$$

By further substituting Equation 3.4 into Equation 3.3 we recover the range error, ΔZ, as a function of object of interest range, Z, at a fixed disparity error, Δd, Equation 3.5.

$$\Delta Z = -\frac{Z^2}{fB}\Delta d \tag{3.5}$$

applying this to Equation 3.1 yields:

$$X' = \frac{Z'(u - cu)}{f} \quad Y' = \frac{Z'(v - cv)}{f} \quad Z' = Z + \Delta Z \tag{3.6}$$

We now formulate point $P' = (X', Y', Z')$ as being the actual stereo triangulated point estimate from a disparity map created with a disparity error of Δd. Our assumption is that as the stereo data and ground truth data are registered, we can calculate any difference in stereo disparity to ground truth data as the Euclidean distance between P and P', This can be recovered via Iterative Closest Point (ICP) registration [111] between the data and ground truth point clouds of an isolated foreground object.

In Figure 3.4 we can see the combination of the ground truth data laser scanned point cloud (white points), the point cloud obtained from a given dense stereo approach (coloured points) and the bounding box annotation supplied with the dataset [34] for the car objects (red lines).

From the ground truth annotation (Figure 3.3) we isolate foreground objects of interest (i.e. cars) in both the disparity point cloud and the laser scanned ground truth point cloud. The bounding boxes are expanded by a fixed ΔD, (0.25m), where the edges of the boxes are defined as $x_{min} = x_{centroid} - width/2 - \Delta D$ and $x_{max} = x_{centroid} + width/2 + \Delta D$ and similarly applied to y_{min} and y_{max}. This is to compensate for any poorly triangulated point positions and ensure they are captured within the bounding box of the relevant object. Maximising the number of points that belong to an isolated object will increase the ICP registration performance.

From the two resulting object of interest point clouds, extracted from stereo

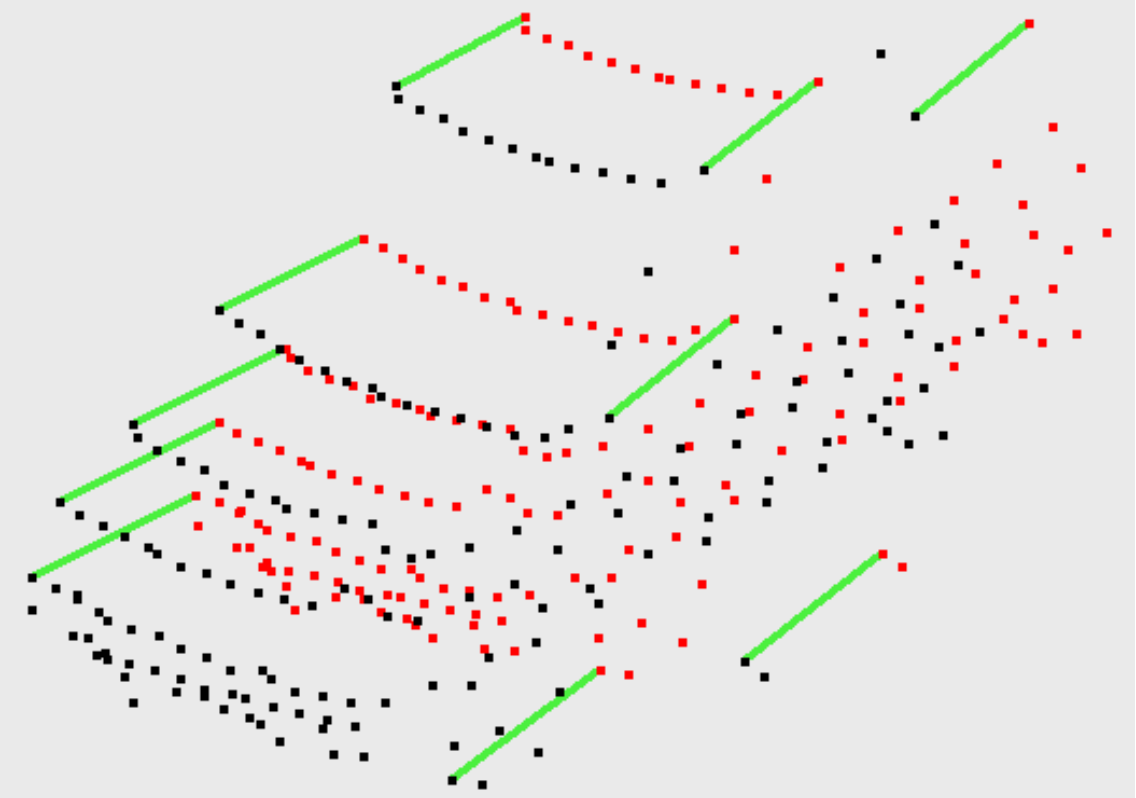

Figure 3.6: Raw ground truth data for a car at a range of 25m (black points). Ground truth data post ICP registration with disparity point cloud (red points).

disparity and ground truth laser scanner, we perform registration using the Iterative Closest Point [111] approach. This facilitates the recovery of the transformation between the two point clouds. Post registration, we can identify the Euclidean distance offset between the two point clouds. This provides us, via ICP, with a global accuracy metric of the stereo correspondence algorithm for the reproduction of the ground truth 3D information for a given object. Furthermore, as each object of interest is extracted at a known depth from the camera, Z, based on the ground truth annotation we additionally have an accuracy metric relative to object range (Figure 3.4). Figure 3.6 shows an isolated point cloud from the laser scanned ground truth data as the black points and the registered version as the red points. Lines illustrate the offset between point clouds which we denote as displacement error. Performing this over the test sequence yields numerous such object-wise accuracy measurements over a range of distances (range, Z) and angle to target (Figure 3.4). The results over the test sequence are presented in Figure 3.8.

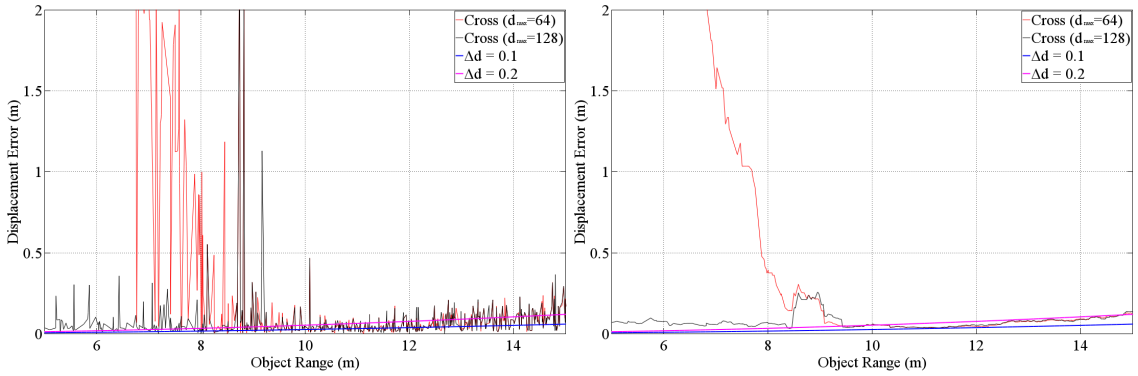

Figure 3.7: Short range accuracy difference between max disparity parameter settings. Left: Raw data. Right: 0.5m smoothing window.

3.3 Results

This study uses the data set available from [34] which is provided with ground truth 3D laser scan data and annotated bounding box information for cars, trucks, pedestrians and cyclists. Specifically the analysis was carried out on two sequences entitled '2011_09_26_drive_0009' containing 89 cars over a sequence of 447 rectified images and '2011_09_26_drive_0023' containing 150 cars over a sequence of 480 rectified images. Data in both sequences is collected at 10Hz. These two sequences were selected due to their high number of objects labeled as 'Cars' providing a large number of data points to process. We model this expected drop off in accuracy over range of a given stereo correspondence algorithm (i.e. Equation 3.5) and plot the theoretical range error for two hypothetical disparity matching errors, $\Delta d = 0.1, 0.2$ in pixels (Figure 3.8). This gives us a base-line expectation of degradation of displacement error against range against which to compare (assuming a constant disparity error over range).

The displacement error, obtained via ICP registration, is plotted for every visible and labelled car in every frame in metres and collated for a collection of algorithms (Figure 3.8). Here we test BM, SGBM, NoMD, Cross, AdptDP, ELAS and Non-Local. From Figure 3.8 we can see a distinct difference in performance in relation to the range of the object from the camera. At short ranges it is apparent they can perform very differently. This can be attributed to the parameters used when tuning the algorithms (e.g. setting BM or SGBM to a limited disparity search range

of 64 pixels artificially limits the minimum effective range, hence the dramatic drop off in accuracy). Increasing this parameter increases the accuracy (decreases the displacement error) performance at short ranges (Figure 3.7).

Algorithms not limited to a disparity search range naturally have improved close range performance (e.g. ELAS and NoMD, Figure 3.8). The rapid increase in the displacement error at small ranges is due to the limitations of ground truth data collection. As the foreground vehicle comes within the minimum range of the laser scanner and the edge of the field of view of the cameras, the two point clouds become patchy and incomplete. This reduces the registration constraints for the ICP registration causing artificially high registration displacement errors (Figure 3.4).

The left image in Figure 3.8 shows the displacement error for the algorithms that performed best in this study. We see that BM, SGBM and ELAS all perform with a $\Delta d < 0.1$px. The collated displacement errors for NoMD, Cross, AdaptDP and NonLocal are seen in Figure 3.8(right). We can see they have a matching accuracy of $\Delta d \approx 0.15$px.

Due to angular resolution of the laser scanner, objects beyond ~ 35m in range have fewer points for the ICP method to match against. While errors of up to 1.5m at a range of ≈ 35m may not sound significant, (Figure 3.8, right) a vehicle travelling at the UK motorway speed limit of 70mph (≈ 31m/s) will cover that distance in a little over 1.1s. Furthermore, the typical width of a car within the data set is ≈ 1.6m which in itself is close to the magnitude of this error.

Figure 3.8 (top left) shows the top performing algorithms in this study (BM, SGBM, ELAS). These three algorithms maintain a low displacement error throughout the test ranges indicating a low matching error, Δd. The greater the matching error the poorer the depth estimation and the greater the resulting displacement error, as seen in Figure 3.8(top right). Interestingly BM and SGBM both perform as well as ELAS, only requiring that the maximal disparity search range is increased to cope with close range parts of the scene. Using the known hardware configurations of the stereo camera we can simulate the expected 3D accuracy given an expected stereo matching error. Figure 3.10 shows range-azimuth plots for the KITTI config-

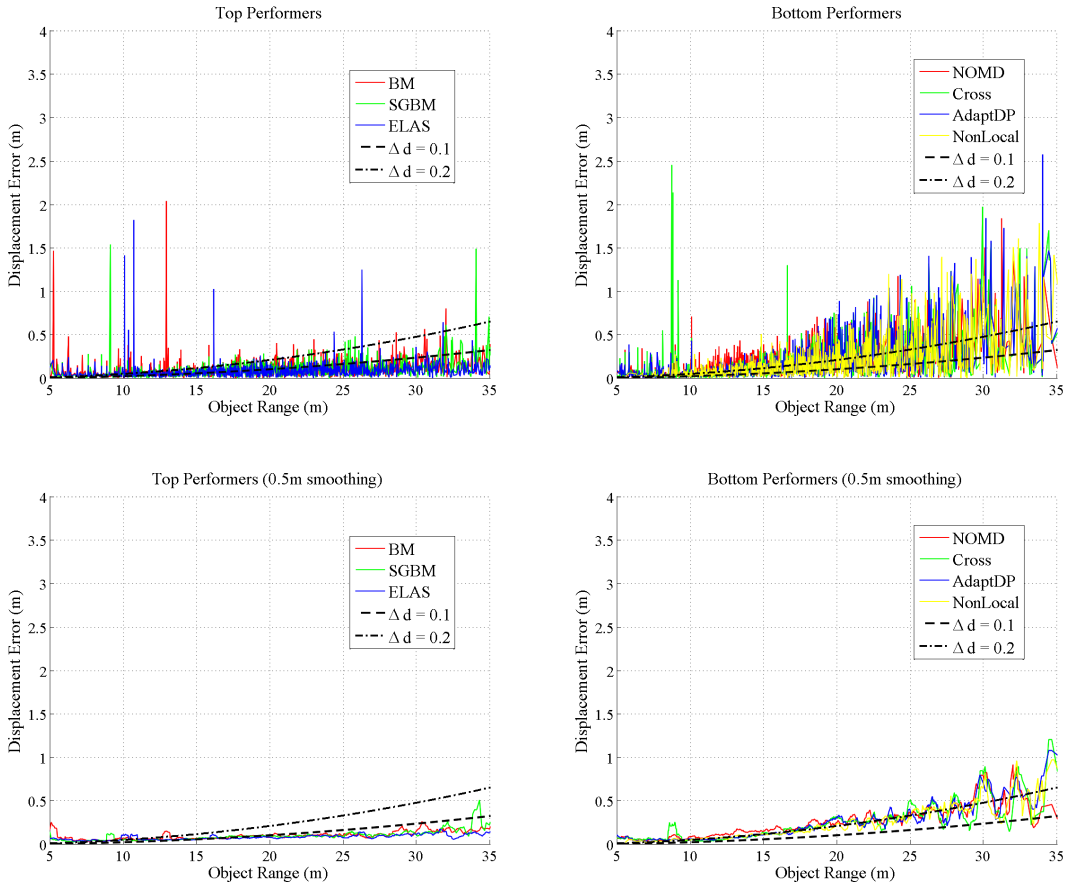

Figure 3.8: Top left: Stereo algorithms with higher accuracy. Top right: Less accurate algorithms in this study. Bottom left and right images have a 0.5m smoothing window applied.

uration at two disparity matching error levels. The accuracy not only falls off with range but also decreases with azimuth as the effective baseline component reduces as a function of $cos(\theta)$ (Equation 3.7). At the extreme of $\theta = 90^{o}$, rays projected from each camera to 3D points at all ranges lie on the same axis as the stereo cameras, therefore there is no parallax observed and no stereo triangulation can be performed. Accuracy as a function of elevation, ϕ, will not suffer in the same way, as rays projected from the cameras to 3D points at all values of $\phi, (\theta = 0^{o})$ are perpendicular to the stereo camera baseline axis therefore a parallax exists to perform the triangulation.

$$\Delta Z = -\frac{Z^2}{fBcos(\theta)}\Delta d \tag{3.7}$$

An alternative way of examining the data is shown in Figure 3.9; here we use

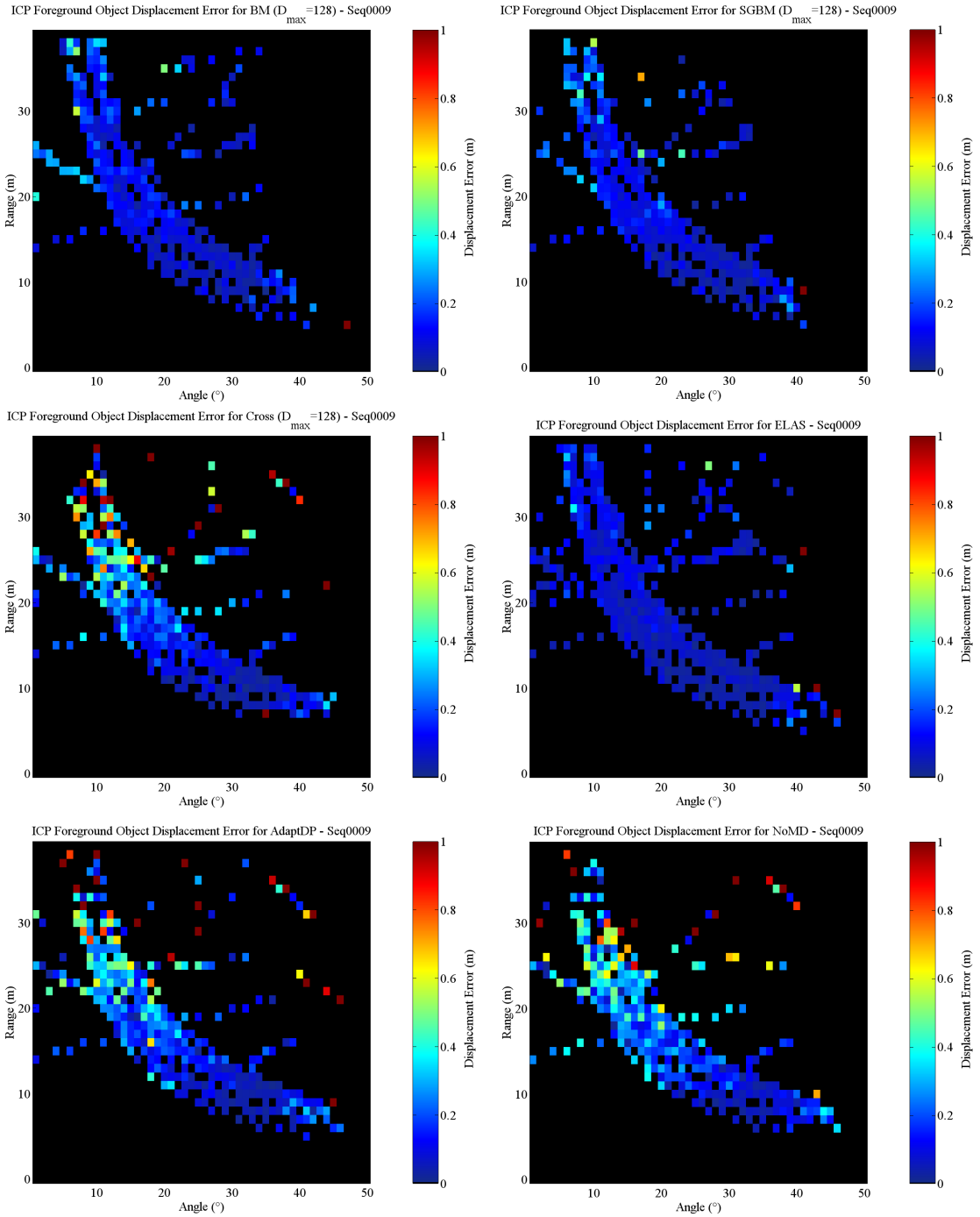

Figure 3.9: The Range-Angle map showing a given stereo algorithm's accuracy coverage of an observed scene. ICP error is clamped to a maximum of 1m to enforce colour scale consistency across all range-angle maps.

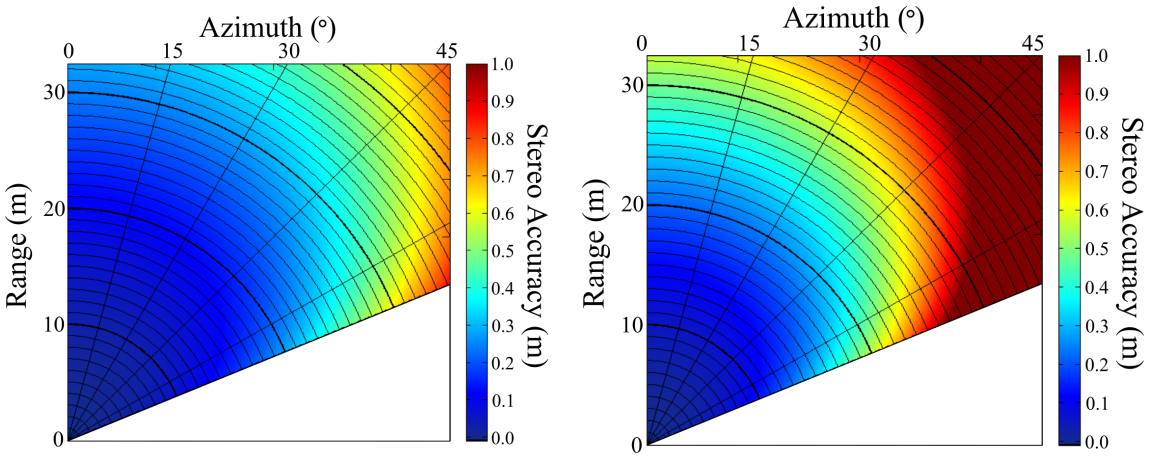

Figure 3.10: Predicted stereo accuracy as function of range and azimuth for KITTI camera configuration. Left: Range error with 0.1px disparity error. Right: Range error with 0.2px disparity error.

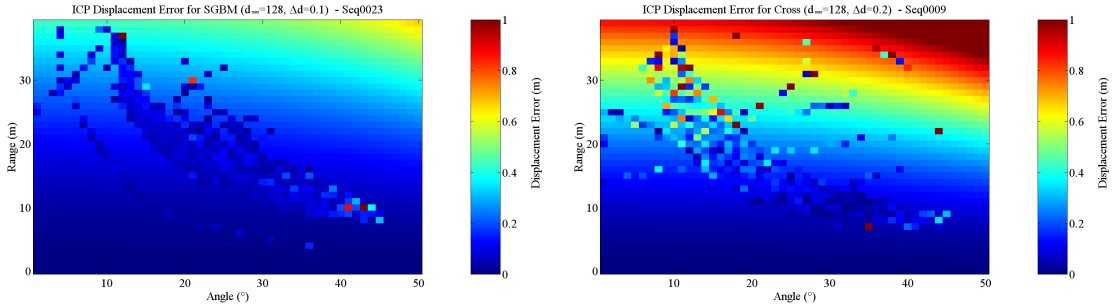

Figure 3.11: An overlay of the stereo accuracy results on top of the stereo accuracy prediction with a given disparity matching error. Left: Dense stereo algorithm 'Cross' results overlaid on theoretical stereo matching error of 0.1px. Right: SGBM results overlaid on theoretical stereo matching error of 0.2px.

an addition quantity to represent the data in 2D. The angle a foreground object is seen, with respect to the centre of the image, is recorded and used to plot the data as a range against angle map with the value in a given range-angle bin being the average computed ICP error. The observed angle is an averaged angle as the reported location of the tagged foreground objects relates to their centre. As cars get closer to the logging platform, their apparent size increases and they occupy more of the field of view; however we only record the observed angle of the centre of the object. Converting the range-azimuth plot to Cartesian and preserving the colour mapping scale we can underlay the theoretical accuracy plots under the real-data plots, shown in Figure 3.11, as illustration as to how close an algorithm is to theoretical 3D point accuracy. Further examples of range-angle plots for dense stereo algorithms against $\triangle d = 0.1$ are shown in Appendix A.2.

Despite these insights, some notable limitations of this methodology relate to the poor reliability of displacement distance at ranges greater than $\sim$ 35m. This is somewhat due to the angular resolution of both the laser scanner and the disparity images, which results in fewer points being used to construct an object of interest at significant ranges, Z.

Overall, from the left image in Figure 3.8 we can see that BM, SGBM and ELAS perform well over a large range whilst Cross, NoMD, AdaptDP and NonLocal are notably seen to perform less well at ranges greater than $\sim$ 15m (Figure 3.8, right). The results demonstrate the effectiveness of ELAS by maintaining a low object wise displacement error throughout the test range whilst not itself being inherently limited to a maximum disparity search range (unlike others, e.g. BM, SGBM).

Whilst the limitation of ground truth quality at significant range mildly affects the methodology we still see a clear quantitative insight into relative algorithm performance.

3.4 Conclusion

Using this novel quantitative evaluation approach for dense stereo algorithms, considering object-wise foreground accuracy in relation to range, we compared a range

of dense stereo approaches providing novel insight into complex urban environment performance. From our small sample of algorithms, we conclude that ELAS performs with the greatest foreground object accuracy throughout the ranges examined in this study. However, it only marginally beats the early attempts of BM and SGBM and only on one aspect. Accuracy as a function of range across all three (ELAS, BM, SGBM) are on par with one another using this assessment technique; the advantage of ELAS is its versatility. BM and SGBM, along with many other stereo techniques, require a parameter to define the disparity search range, therefore potentially limiting their effective minimum range. ELAS, being feature driven, does not require this predefined limit. In implementation this limitation of a defined disparity search range could be overcome by using a variable search range that is inversely related to vehicle speed. At high vehicle speeds there is a greater need to process images faster, in order to act accordingly, therefore a smaller maximum disparity search range is favoured while the opposite is the case for low vehicle speeds.

What is also clear from this work is that while there has been vast amounts of stereo vision development and evaluation, there is very little in the way of assessing the effectiveness of dense stereo in a variety of environmental conditions such as rain, snow, or in low light levels. At typical urban driving speeds of $\sim$ 13.4m/s (30mph) the KITTI data [34] uses shutter speeds (exposure time) of a maximum of 2ms, equating to the camera traversing 2.6cm during the sensor integration period. In lower light conditions a 20ms exposure time is a reasonable figure, especially for smaller aperture automotive optics, which would cause a positional uncertainty of 26cm over the period of image integration, a figure comparable to stereo error metrics calculated in Section 3.2. Sky luminance can vary from $10,000$ candela per square metre (or 'nits') on a clear day down to 100 on a heavily overcast day [112], assuming a fixed aperture and sensor gain this would vary the image integration period from 2ms to 200ms causing a positional uncertainty of 2.6m at speeds as low as $\sim$ 13.4m/s (30mph), resulting in far too much image blur and uncertainty, therefore sensor gain must be increased to obtain acceptable images. High values of sensor gain cause extra image noise and therefore will change the effectiveness of a given stereo algorithm.

3.5 Future Work

As discussed previously we have yet to evaluate low light stereo imaging which would be a vital requirement for any commercial deployment of a stereo based algorithms and therefore be the subject of a future investigation. Exposure time is not the only way to control image intensity, sensor sensitivity (gain) or aperture both affect the image brightness. A variable aperture is not an ideal solution as it involves mechanical moving parts. Sensor gain increases the image brightness at the cost of image noise, which will also negatively impact the performance of stereo matching algorithms and SVO. Quantitative results for further blur analysis could be obtained by simulating images. However, generating photo realistic blur is not a trivial task [113] and, as previously discussed, the results obtained on simulated images as compared to those seen on real-world data sets can vary greatly [36, 102]. Collecting identical data sets consisting of a moving stereo-camera, through a consistent scene, with multiple exposure settings would prove useful for quantitative assessment of dense stereo and SVO algorithms. It would however be very challenging on a road going vehicle, as retracing the exact camera path would be very difficult. A platform constrained to path, such as a tracked train system, would be required.

3.6 Summary

We have demonstrated that the dense stereo algorithms ELAS [3], BM [50], and SGBM [5] all have similar performance when it comes to reconstruction accuracy with the KITTI [34] data using our novel object based evaluation approach. We have shown theoretical accuracy forecasts as a function of range and angle due to apparent baseline changes for observations off axis. Acknowledging the significant performance difference between static indoor scenes and dynamic environments we propose future work to examine the impact motion blur and noise has on dense stereo reconstruction and SVO. We conducted an initial qualitative experiment as a proof of concept that shows dense stereo algorithms can function under basic artificial blurring, highlighting the potential for future research on the problem.

Chapter 4

Structure from Motion

The following section will explore the theory behind Structure-from-Motion (SfM), the process of reconstructing a scene from a single moving camera. It will detail our SfM approach for creating a uniform sparse point clouds, performing an essential filtering stage, optimisation and refinement processes through the use of a dynamic bundle-adjustment window, and our novel online densification stage. We specifically optimise the process to the camera configuration found on our data logging platform.

4.1 Review

Processing steps of SfM, Visual Odometry (VO) and SLAM are all very similar in nature by the fact they are all concerned with camera position, or pose, within a scene and the structure of the scene. The various methods have different priorities and objectives. Typically SfM, as the name suggests, is primarily focused on recreating the structure of the scene observed from a single moving camera. SfM approaches tend to produce high quality 3D data at the cost of processing time. VO and SVO are concerned with calculating camera pose at higher frame rates usually at the expense of fewer feature points being tracked, albeit strong features, and therefore less dense 3D model outputs [65]. There are three main stages to all the approaches 1) Find features in two, or more, frames. 2) Estimate camera motion from matched 2D features and calculate essential matrix to decompose into pose information. 3) Refine and optimise camera poses to minimise reprojection error

of 3D position estimated projected back into 2D images. Very recently there has been a shift away from feature based tracking, matching, and essential matrix estimation for pose extraction. Work by [29] uses direct image intensities to estimate camera pose change by using an initial estimated depth map which is then refined with subsequent pose estimations. Their approach achieves reasonably high quality semi-dense 3D models with the majority of the depth information being found, but limited to image edge regions. It performs well at high frame rates on a single CPU, however in order for their implementation to quickly converge on a solution for pose estimation of new frames, it performs best with small changes in the scene to minimise the amount of image offset in a given time.

4.1.1 Typical SfM Process

Much of the previous work in SfM/SLAM uses low resolution high frame rate [92,94,114,115]. In this work we focus on using high resolution images captured at a lower frame rate of around $5-10$Hz. As briefly outlined in the previous section there are two main schools of thought when it comes to performing SfM. The oldest and most well established approach is to use feature points. This entails using one of the many feature detectors and feature descriptors (e.g. SIFT, SURF, ORB, etc. [116]) to extract common correspondence points between image pairs or groups of images. After matching the 2D features the Fundamental matrix, F, can be computed. F describes the relationship between corresponding points in stereo images, or in this case two images captured from a single camera at a different spatiotemporal locations. The inherent geometry of two images observing the same scene, referred to as epipolar geometry, allows for an elegant formulation to describe the estimated location of correspondence points between image pairs using a homogeneous coordinate system. Figure 4.1.1 illustrates the epipolar geometry of two images (either stereo or from a moving camera). In this example the observed point 3D world point X is observed to be at 2D locations in each image at x and x'. Point C is the principal point or centre of the camera that passes through the optical axis and a distance of f (focal length) away from the image plane. The shaded region is referred to as the epipolar plane which projects to a epipolar line in 2D image space as defined by Fx.

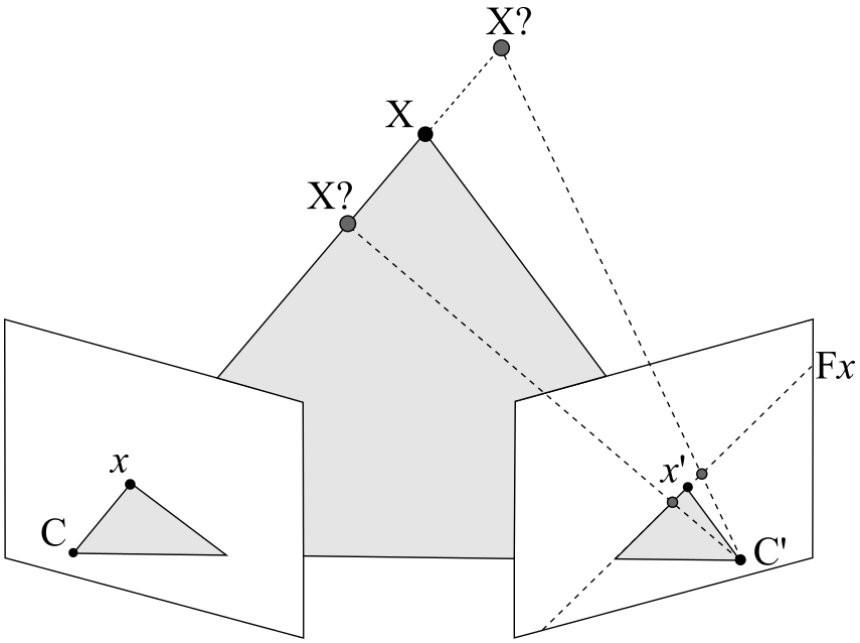

Figure 4.1: Illustration of epipolar geometry and how a point observed in two images can be related using the Fundamental Matrix, F.

$$x'Fx = 0 \tag{4.1}$$

Under ideal conditions world point X projects to x and lies along the line Fx so to satisfy Equation 4.1. However in the limit of image noise, pixel quantisation, optical aberrations, and even false matches, this is often not the case therefore it must be solved by minimising Equation 4.1 for all matched points in the images. Trying to minimise Equation 4.1 for all points can result in incorrect values for F skewed by incorrect point correspondence matches, alternatively an iterative optimisation method, referred to as RANSAC (random sample and consensus) [100], is employed to find a value for F that best satisfies a model for a random set of points, the inliers. As such, RANSAC is a non-deterministic algorithm that fits a model to data with a given probability that typically increases with increased iterations. A deterministic approach to finding F is known as the Eight-Point Algorithm [19, 117].

$$E = K'^{T}FK \tag{4.2}$$

From F we can compute the Essential Matrix, Equation 4.2, where $K' = K$ for a

moving camera with constant intrinsic properties, otherwise K, K' typically refer to the individual intrinsics of left and right cameras respectively for a stereo head. This Essential Matrix encodes physical properties of the cameras, their intrinsic parameters from (K) and their external positions relative to each other, or their poses. To obtain the relative pose of each camera we use Singular Value Decomposition (SVD) to decompose the essential matrix (E) into rotation and translation matrices R and T respectively [23]. This step actually yields multiple solutions (R, R', T, T') with only one combination of $[R|T]$ being the correct solution for the camera positions. Selecting a subset of the the features and testing combinations of R and T by triangulating the features to 3D world locations and testing their position relative to the cameras we can determine the correct camera positions by simply counting which subset and combination of R and T results in the most number of features being triangulated in front of the cameras. The process of detecting features, match, compute poses is performed on all subsequent frames to construct the 3D nature of the scene and recover the motion of the camera. Each estimated camera pose will have an associated error on both rotation and translation components caused by feature position error from image noise, the RANSAC stage of estimating the fundamental matrix and the intrinsic matrix values K. With the addition of each new frame and processing of SfM stages, the estimated camera pose accumulates this error and drifts away from the true camera position. In order to reduce the amount of drift a process commonly known as Bundle Adjustment (BA) is used [20]. BA is a large scale optimisation problem that is used to adjust the positions of the cameras so as to minimise the reprojection error of the 3D points into the 2D image positions in each camera from which the point is observable.

One well known limitation of the SfM process is that $\| T \| = 1$, that is the distance between any two cameras in isolation is not recoverable from the essential matrix alone [23]. The impact of this means the real scale of the 3D scene is also not recoverable directly from SfM. This is one of the main advantages of dense stereo processing; the known baseline constraint of the stereo camera configuration allows for correct scaling of the triangulated 3D points. The work of [59] manages to compute world-scaled SfM from knowing the height of the camera above the road

surface. By creating an initial 3D model via an SfM process, of normalised scale, they then fit a plane to the road surface and estimate the intersection of the camera and road surface and scale this dimension to the previously measured height. While demonstrating to be very accurate and reliable [59, 63] on this automotive dataset, it will likely suffer when encountering rough terrain or when a road surface is not visible, for example when manoeuvring in a car park where very little surface is visible, obscured by other vehicles. Another unknown is the sensitivity to changes in height, as passengers and cargo are loaded a vehicle's suspension will compress, lowering the height of the camera too. Alternatively a scale factor may be obtained by using GPS to determine the scale of the motion of the camera, however many civilian global navigation satellite systems (GNSS) do not provide positional data to a sufficient accuracy. The US GPS and the Russian GLONASS both achieve an accuracy of around 5m [58], the new European Galileo system (still being deployed as of January 2017) is capable of achieving sub metre precision [118]. While it is possible to achieve positional accuracy on the centimetre scale [119] with GPS, it requires specific scenarios and is far from mature enough to be exploited at this stage. The reliance on space-based RF signals is probably never going be robust enough in all situations to provide a location precision comparable to visual navigation solutions for small scale localisation. This is one of the challenges we address within this work, creating scaled SfM results from video data only with little constraint on camera placement.

4.2 Proposed SfM Process

The approach taken in this work was to use temporally sparse but spatially dense data, therefore sampling images at around 7.5Hz but at high resolutions of 1280×960px. The motivation for this approach is to minimise the triangulation error and therefore the camera positional error. As with dense stereo imaging, the resolution, along with baseline, has a significant impact on the 3D triangulation accuracy. By decreasing the frame rate we also increase the effective baseline of a single moving camera at a given speed. The increased baseline and higher precision of the 2D

point locations should allow for lower 3D triangulation error. The placement of the camera is an important factor in our processing methodology. By placing the camera on the side of the moving platform, facing outwards, we can approximate a moving single camera at different positions to a stereo camera pair, thus allowing for a multi-view stereo approach to create dense depth maps, Section 4.2.2.

We employ a two stage approach to 3D reconstruction. Stage one is an optical flow driven sparse bundle-adjusted SfM pipeline, Section 4.2.4. Stage two is a stereo rectification process used to enable processing using dense stereo algorithms, Section 4.2.5. The following sections detail the sub-processing steps within the two stages.

4.2.1 Data Collection

From the outset it was clear a custom dataset was going to be required as many existing datasets either focused on forward facing stereo cameras or roof mounted omnidirectional imaging system. Existing stereo datasets suffer from the problem this work aims to overcome, that is one of coverage around a vehicle. The available omnidirectional or fisheye datasets typically capture 360° around a vehicle, allowing for full 3D reconstruction of the environment surrounding the vehicle. However, they often suffer from low angular resolution, thereby reducing the accuracy at longer ranges compared to standard pinhole-like camera models. The most recent dataset [120] contains multiple-baseline stereo images in the form of a forward facing triple-camera configuration accompanied by multiple fisheye cameras around the vehicle providing surround view imaging. Having only very recently being released it was unavailable for use in this work. As a result we aimed for a hardware configuration that minimises the number of cameras and maximises the potential spatial mapping resolution.

4.2.2 Hardware Configuration

A custom data collection platform has evolved throughout the project. Early versions were centred around using forward facing stereo cameras mounted to a robotic platform (Pioneer 3-AT) with data being logged to a laptop (Figure A.16). This al-

lowed for frequent indoor testing and early algorithm development. As the research progressed, a wider range of more representative data of outdoor environments was required. The robotic platform was suitable for outdoor use, however collecting data outside quickly highlighted an issue with its size. The wheelbase of the data logging robot being only 268mm and using wheels of 222mm in diameter meant that it was particularly susceptible to mildly rough terrain. Whilst it could easily traverse gravel, tarmac or grass, the small dimensions exaggerate the profile of the terrain it traverses. This caused large amounts of inter-frame movement, in both translation and rotation, which caused problems with image quality in the form of motion blur. Migrating the data logging system onto a vehicle was vital for capturing usable data. See Appendix A.5 for photos detailing the evolution of capture hardware configurations.

The final version of the data capture platform used two Point Grey USB 3.0 Flea3 cameras with 8mm fixed-focal length lenses, in a forward facing stereo configuration with a baseline separation of 0.16m, two side facing (left and right) Point Grey Firewire Flea2 cameras with, wider angle, 4.4mm fixed-focal length lenses, mounted back behind the stereo cameras. Camera synchronisation was controlled via an external 5v trigger signal supplied from an Arduino microcontroller. For the size, weight, power, and cost considerations we used a low power Intel i5 based system to log the data to a 2.5" HDD (hard disk drive), it was evident quite early on that when writing data to the HDD it was not capable of keeping up with the raw image data rates. The system was upgraded to use SSDs (solid state drives) configured in RAID0 (Redundant Array of Independent Disks, 0 indicating the mode of striped data read/write where data, at the hardware level, is split between each drive, therefore achieving approximately 2x the read/write speeds) to further improve the recording bandwidth to cope with the four streams of uncompressed images of 1280\times960 px at 7.5fps.

4.2.3 Data Capture

The primary data used in the rest of this work was collected at various locations around Durham, UK. The logging platform was mounted on top of a Mitsubishi

i-MiEV electric vehicle, Figure 4.2. Recorded environments were typically urban roads with mixtures of pedestrians and vehicles. The typical speeds for sections of the data used in reconstruction were 10-15mph.

Figure 4.2: Data logging pod mounted on our Mitsubishi i-MiEV electric vehicle. Left, schematic view from above. Right, Parked near Durham Cathedral, the site of many of our data collection activities.

4.2.4 Sparse SfM Using Optical Flow

The sparse SfM stage forms the basis of the reconstruction from the monocular side-facing cameras. It is imperative that this stage performs well and is able to estimate the camera pose to a high degree of accuracy in order to perform stage two. In general there are two approaches to SfM global and local. A global approach traditionally extracts key features for every image in a given set of images then matches every frame to every other frame. This approach has been shown to produce very high quality 3D points and camera poses [37]. However, it has the obvious downside of being restricted to requiring all frames prior to processing which is not possible if the system is to generate 3D information as the vehicle transits through a scene. In this project the aim is to construct accurate 3D models as the vehicle transits an unknown environment; for this we use the local method of SfM. Local reconstruction essentially follows the same process of feature extraction (Section 4.2.4.1), matching (Section 4.2.4.2) and bundle-adjustment (Section 4.2.4.5)

to optimise the 3D points and camera poses. In this case the system just optimises over the last n-frames, where n can be determined by several means. A simple approach would to use a fixed value of sufficient size i.e. ~ 10. Too few and the optimisation becomes very localised to the last few frames which can still result in significant drift. Too large a window over which the bundle-adjustment is performed results in a poor performance with the points that are no longer visible in current frames being optimised when no further new information is being obtained. The impact of the bundle adjustment window size is discussed and tested in Section 4.2.4.7.

4.2.4.1 Features

Initially 2D features were extracted using traditional feature point detectors like SIFT, SURF, ORB, etc. [116], these are well established feature detectors that performed well on our data. However, some detectors are prone to clumping features in some regions of the image. This has the undesired effect of weighting the feature locations to specific regions of the image. Figure 4.2.4.1 illustrates this effect where the foliage on the left of the image has approximately as many features associated with it as the rest of the image, the feature points are also approximately same range $\sim$25m as measured from Google Earth [1]. Since 3D triangulation point accuracy is a function of range, the further away a point is the less effective the optimisation process becomes, therefore having a highly dense region of 2D image features on objects that are far away dominates any optimisation processes, particularly the bundle-adjustment process used. To avoid non-uniform feature distribution we use the simple and fast feature detector of [121] with an implementation that selects the strongest features outside of a minimum radius from neighboring features. For the size our input images, 1280×960px, we typically use nearest-neighbour distances of 20-40px. A smaller distance results in a greater number of points that improves reliability at the cost of processing time. These feature points do not necessarily make for good feature descriptors using such methods as SIFT, SURF, ORB, etc. [116], thus another approach must be used in order to match features points across subsequent frames, we therefore use the optical flow approach of [122].

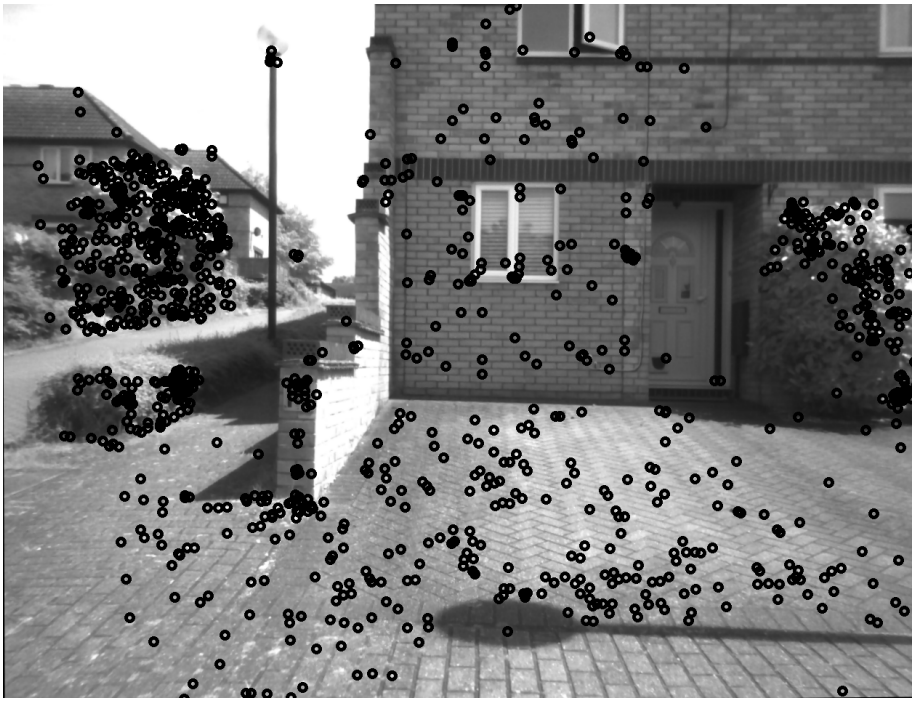

Figure 4.3: Non-uniform distribution of SURF [4] feature detector points. Every 2^{nd} point drawn for viewing clarrity.

4.2.4.2 Matching

The matching of feature descriptors between frames is susceptible to matching one-to-many in the limit of pattern repetition or noise [123] (i.e. a given feature point may correspond to multiple similar features). This can be a particular issue with uniform repeating patterns, as shown in Figure 4.2.4.2, where the bricks and herringbone driveway form similar repeating patterns. In these examples the ratio of feature matching score between the best and second best match is > 0.98 meaning they are difficult to separate in descriptor space alone. The Lucas-Kanade Optical Flow (OF) method [122] performs a least squares fit approach to finding the flow vector. This process is applied to each input feature point and results in an an individual output flow vector for every input point, therefore providing a direct 1-to-1 matching of 2D features between two consecutive frames; this singular mapping is important for the later stages of the SfM processing chain so as to quickly match features across a set of frames in order to optimise them in a bundle adjustment manner. From the output of the optical flow we use the provided quality metric to perform a first pass filter to remove any flow matches that fail to meet a specified

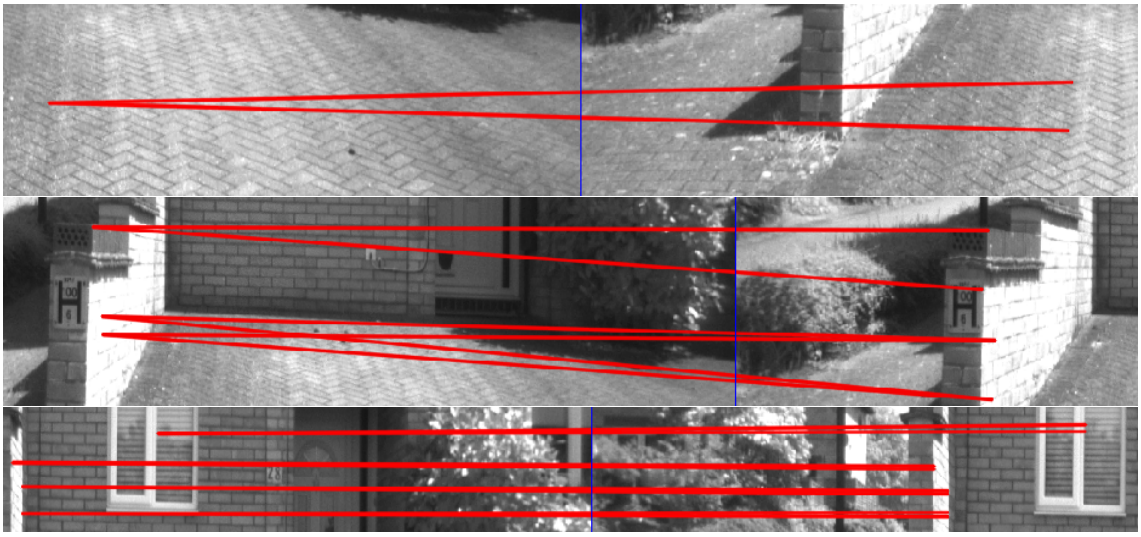

Figure 4.4: Examples of one-to-many feature matching using SURF descriptors.

threshold.

Optical flow performance is controlled by several variables, window size, matching error threshold, number of iterations, and initialisation position of expected flow. Window size is the size of the region around a feature point that is used to perform the matching. The error threshold is defined as the L_1 distance from matched patch to feature point origin divided by number of pixels in the flow window. If we know an approximate flow vector we can prime the search in a given region, therefore requiring fewer iterations in order to converge on a match.

In order to determine the best window size and flow error threshold, a brute force approach was used. We selected five different sequential image pairs that are typical of the scenes observed. For each image pair the optical flow window size was varied from 5px to 101px in steps of 4px (an odd number is used for the window size to ensure there are an equal number of pixels either side of a given feature point) and the flow error threshold was varied from 3 to 60 in steps of 3, creating a matrix of 25×20 each with a unique combination of window size and error threshold. For each entry in the matrix two metrics were calculated, the first being the number of feature points used in the bundle-adjustment process (detailed in Section 4.2.4.5), Figure 4.6 right, the second being the mean reprojection error post-BA, Figure 4.6 left. Collating the results from all test images, Figure 4.8, we calculate the median value in each cell for each window and error threshold tested, Figure 4.7.

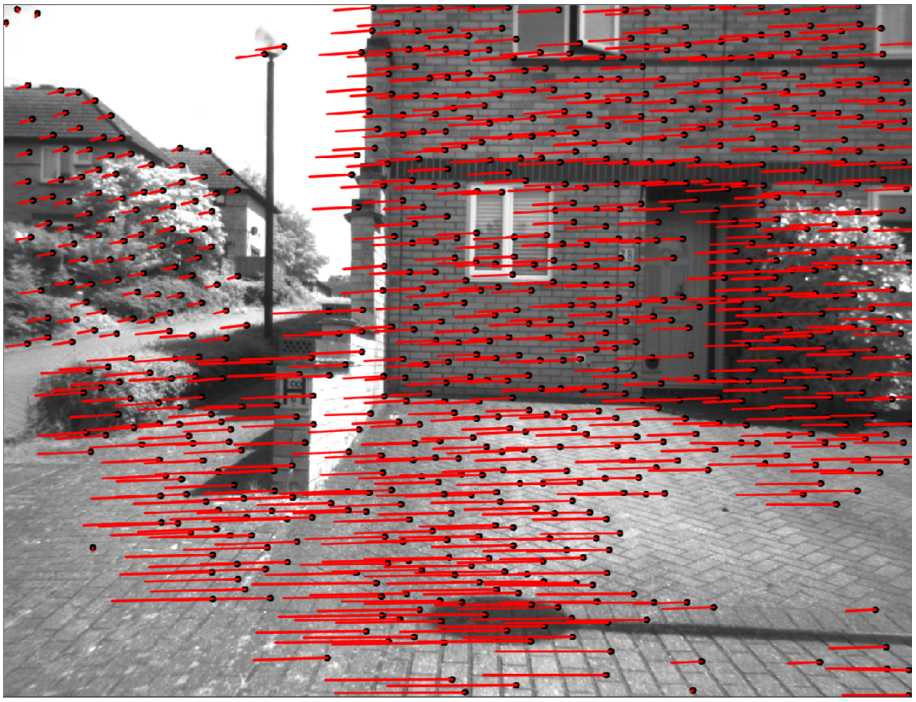

Figure 4.5: Features tracked using sparse optical flow. Black dot indicates the feature point start location, Red line indicates direction and magnitude of motion.

We observe some interesting structure within the results. The number of successful points, Figure 4.7 right, tracked with optical flow that achieve the desired minimum error threshold exhibit banding effects at different window sizes. The source of this is believed to be due to the pyramidal implementation of the optical flow algorithm processing at different image scales in a course to fine tracking scheme. Our image dimensions are 1280×960px, performing pyramid downsampling reduces the image size at each step by a factor of two in each dimension, therefore after one iteration of downsampling the image is 640×480px. After four pyramid steps, a reduction of 2^4, the image dimensions being processed in the course steps of the OF algorithm are now just 80×60px, therefore the OF window size of 61px can not fit within the minimum dimension of the downsampled image. The next, more subtle, banding observed in Figure 4.7 is at a window size of 29px, that being the largest window that will fit within the minimum dimension of the next level of the pyramid downsampling, 40×30px. The conclusion is that the optimal window size in our case is one that is just smaller than the minimum dimension of the downsampled image in the image pyramid. The second metric to analyze is the mean

Figure 4.6: Results from image pair 1 in Figure 4.8. Left: Mean reprojection error in pixels (to 2 decimal places) after BA. Right: Number of valid tracked features used in BA.

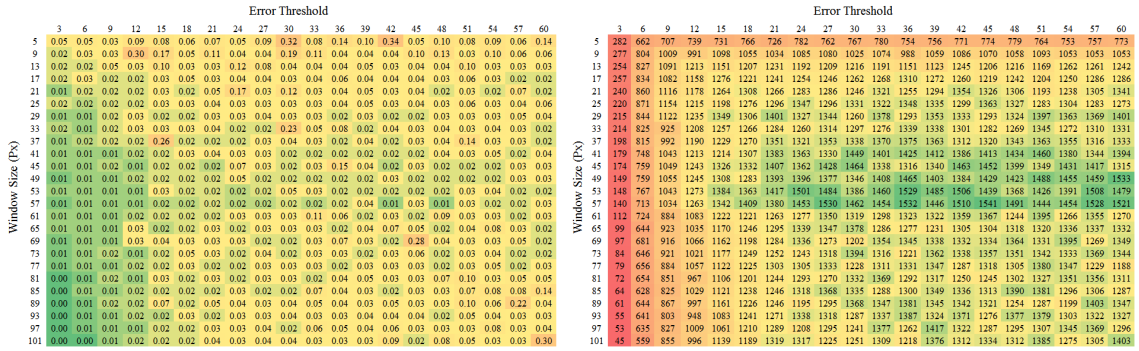

Figure 4.7: Median values from all test cases. Left: Mean reprojection error in pixels. Right: Number of valid tracked features.

reprojection error post-BA, Figure 4.7 left. Here we observe a similar improvement around the same window size from the first metric at 57px, however in this case we do not see the similar pattern around a window size of 29px. From this we can conclude that, in our case, an optimal window size, W is 53px$\leq W <$ 60px. We can also see that the error threshold is less important when using an OF window of this scale. The variation in feature point count, for error thesholds ≥ 15, remains within approximately 15% with the mean reprojection error remaining consistantly below 0.1px. See Appendix A.3 for raw charts used in Figure 4.8.

4.2.4.3 Motion Extraction

After the frame-to-frame 2D features have been matched the camera motion must be estimated. As outlined in Section 4.1.1, this can be obtained by calculating the

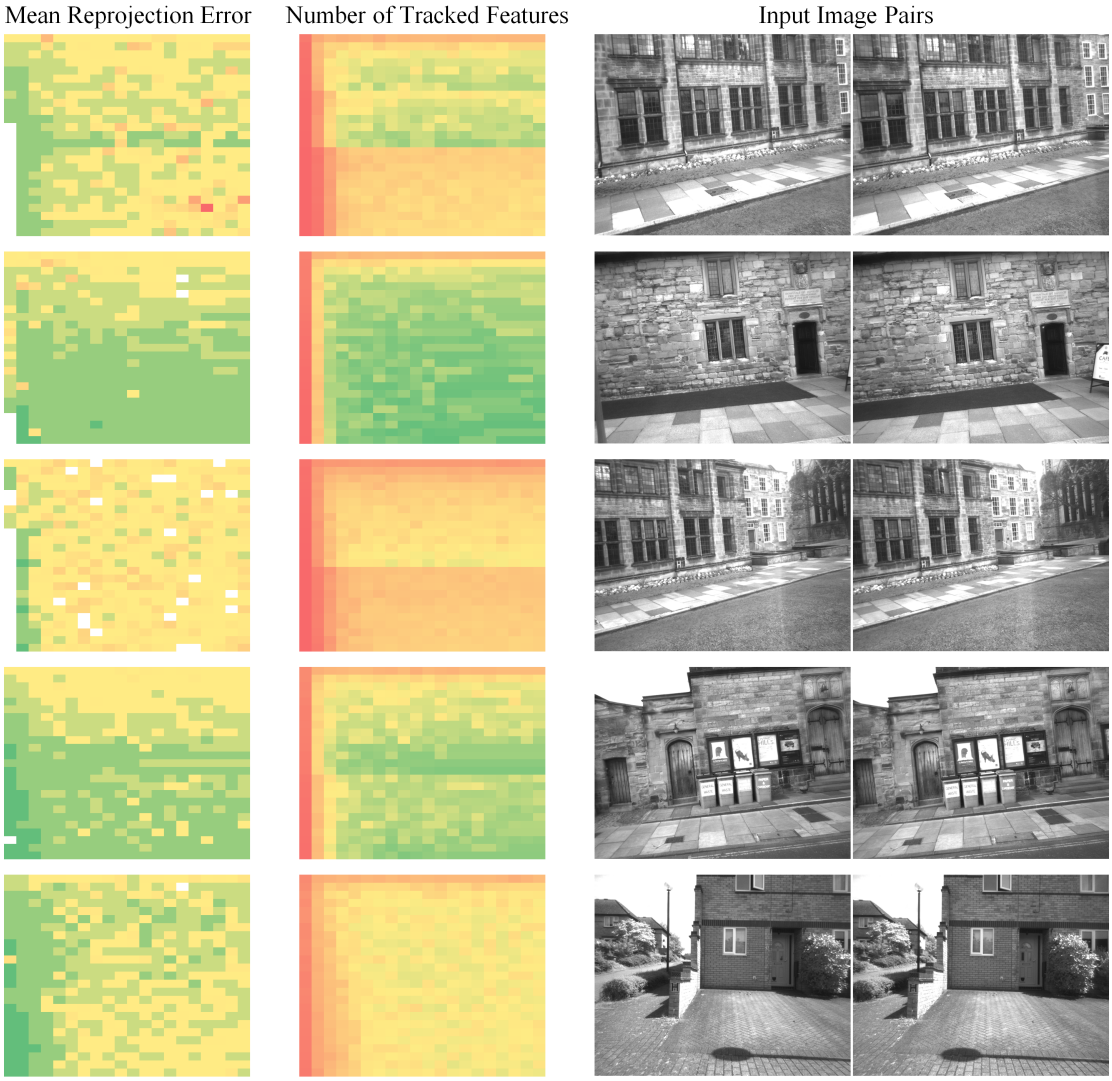

Figure 4.8: Heatmaps for empirical OF parameter optimisation. Each heatmap column has normalised scaling for illustrative comparison.

SVD of the Essential Matrix, E. Using the method and notation of [23] we have the SVD of E being Equation 4.3 to extract the position of the second camera relative to the first.

$$E = U \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} V^T \tag{4.3}$$

$$R_1 = UWV^T \qquad R_2 = UW^TV^T \qquad T_1 = -U(:,1)^T \qquad T_2 = U(:,1)^T$$

$$P_1 = [R_1|T_1] \qquad P_2 = [R_1|T_2] \qquad P_3 = [R_2|T_1] \qquad P_4 = [R_2|T_2]$$

Four possible solutions exist for the transform between the camera positions where the images were captured. To determine which combination of R and T is correct a simple test is performed where a 50% subset of 2D point matches are selected and triangulated into 3D space using each combination of $[R|T]$. By counting the number of points that are triangulated in front of each camera, we can easily determine the correct $[R|T]$ pair as the one resulting in the most. It is important to note at this stage T is normalised and absolute camera position is an unknown from SfM alone.

4.2.4.4 Triangulation

The minimum components of SfM have now been computed to allow for computation of 3D world points from pairs of 2D image points.

Due to imperfect flow tracking, camera calibration, pixel quantisation, and pose estimation, projecting 2D image points into 3D does not usually result in the correct position being recovered that created the points in image space. Consider the example in Figure 4.9 where two 2D image points (x and x') are viewed from two different locations with an estimated transform between the two image planes. A vector is projected from the camera centre outwards through the 2D point observed

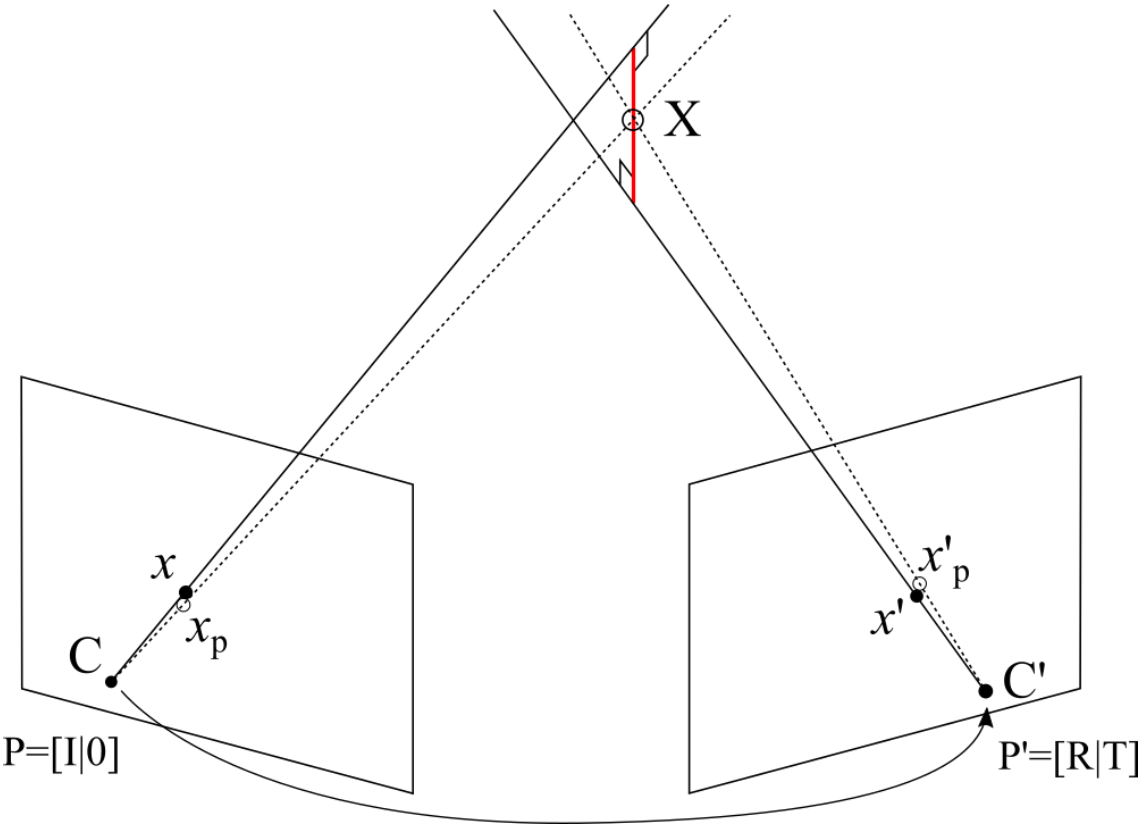

Figure 4.9: Projection of 2D points x and x' into 3D space point X.

in that image; under perfect conditions vectors from every camera passing through the matched 2D point in their respective images would intersect at a point in 3D space corresponding to the observed point X. However, typically the largest source of error is the pose estimation, with the rotational component causing these vectors not to intersect. Therefore in order to estimate the 3D position for two matched 2D points the mid-point of the line orthogonal to both projection rays is used for the 3D triangulated point, Figure 4.9. The estimated 3D point X can then be projected back into the 2D image space, (x_p and x'_p) Figure 4.9, providing a metric for 3D point triangulation quality by calculating the L_2 norm between tracked features, x, and 3D points projected to image space, x_p, this is referred to as the reprojection error. The lower the mean reprojection error the better the estimation of camera poses.

4.2.4.5 Refinement

Using the reprojection error metric we can use this as a goal function for optimisation of camera poses. Bundle Adjustment (BA) is a common approach to optimisation of several different aspects of SfM [20, 23, 38, 90, 124–128]. Given a collection of 3D points observed from different viewpoints and their corresponding detected 2D points in each image, BA can simultaneously refine 3D coordinates, the camera poses, and the optical properties of the camera system. In our case we have calibrated the camera and found the optical characteristics of the camera [23, 129–131], thus removing the need to solve for this when performing BA. In this work we utilise [124] for performing the BA process.

BA is generally performed in two modes, local or global. Global bundle adjustment typically requires matching feature points across all frames in the sequence prior to processing. This approach is typically used in the non-real-time reconstruction found in such packages as Photoscan and the work of [71–79]. As we use Optical Flow (OF) it makes for difficult matching across, for example, the current frame and one taken several frames ago where the overlapping regions' 2D image distance is very large. Optical flow failure can commonly be attributed to large inter-frame motion or due to significant viewpoint changes, resulting in the optical flow reference patch being sampled from a significantly different perspective than the current frame. Instead we use the fact that OF only has a 1-to-1 matching scheme and the frame-to-frame motion is small with a little perspective change allowing for strong OF tracks. From the 1-to-1 matching we create chained-lookup tables to very quickly index any point, and all associated historic matches, over all frames in which it was seen.

Figure 4.10 demonstrates the flow-lookup process that allows for rapid matching of features from the current frame to all previous frames. The maximum value in the lookup array is the length of all the features in the previous frame. This example limits the number of features to 12 for illustration. To find all the flow features associated with e.g. feature(1) in frame(4) we simply go to the index of features in the previous frame pointed to by a lookup table, in this case feature(1) frame(4) maps to features (1, 3, 5, 8) in the previous frames, highlighted by the same blue

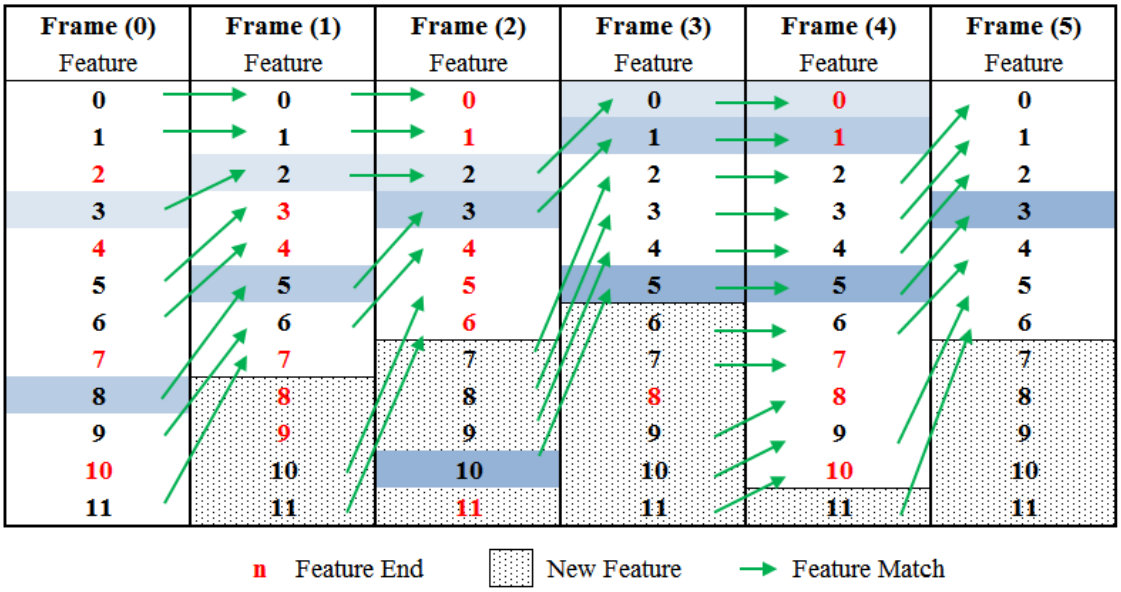

Figure 4.10: Optical flow historic lookup. Red feature index shows features that failed to track to the next frame. Black are valid feature indexes. Dot-shaded cells indicate new features in a given frame. Blue shaded cells indicate flow history of example features. Green arrow indicates flow matches from frame to frame.

shaded cells.

The flow based lookup enables for rapid building of a bundle adjustment data structure, enabling our add-optimise strategy where new frames, from new camera poses, and associated tracked features are added to the feature tracking list and associated with optimised 3D points. We use an adaptive temporal window for bundle adjustment in order to optimise the pose and the triangulated 3D point coordinates; this maintains the scale of subsequent frames in the SfM process to adhere to the scale enforced by the initial frame pair.

The adaptive bundle adjustment window is calculated by determining the oldest frame which contains only features that receive no further updates. The pose of the start frame of the bundle adjustment window is considered to be fixed in space and therefore is our reference frame that ensures the bundle adjustment does not move the camera-group to be bundled away from the previously calculated pose.

Figure 4.12 illustrates how the adaptive bundle adjustment window works; In this example, when frame-2 is captured two features persist from frame-1 and one feature from frame-0. The 'black' feature in frame (0-2) receives new information in frame-2 therefore the current BA window extends from frame-2 back through to

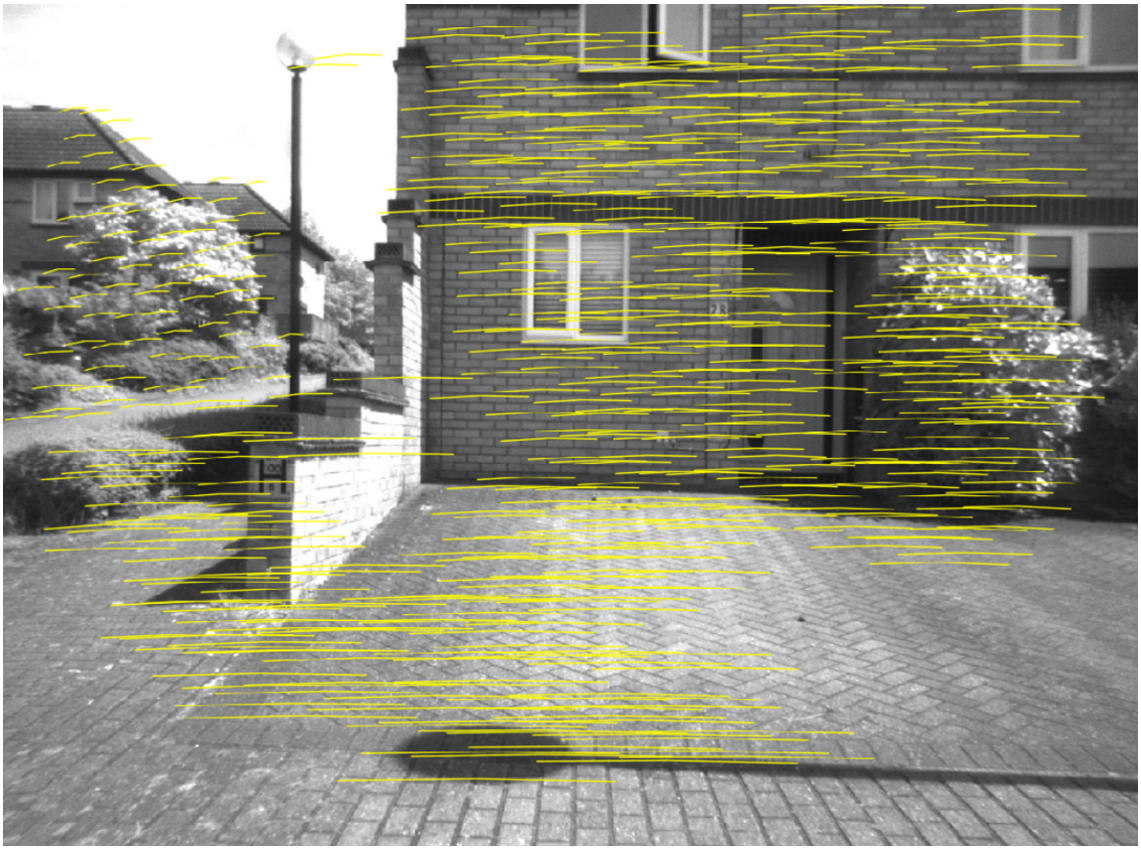

Figure 4.11: Flow features tracked over three frames and indexed using dynamic lookup tables.

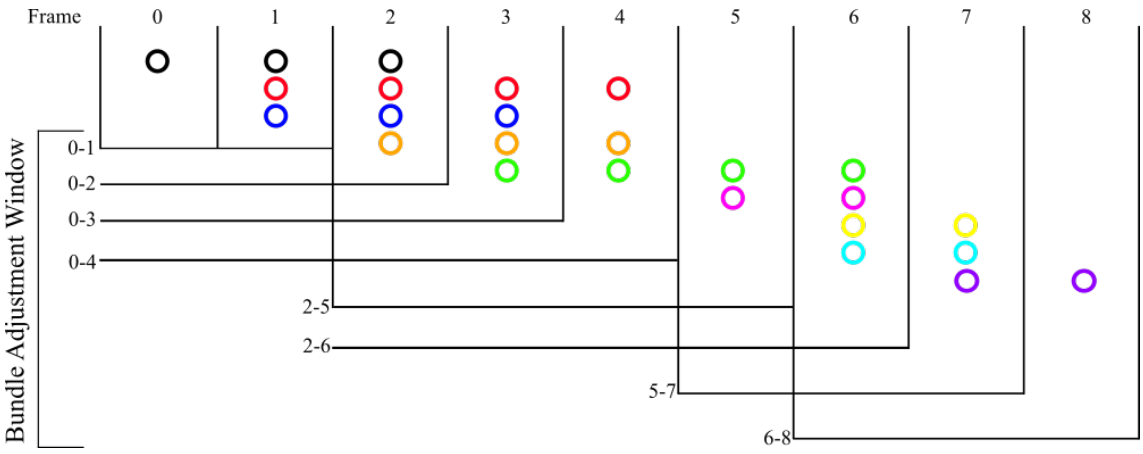

Figure 4.12: Adaptive bundle adjustment window scheme. Circles represent features, columns represent frames.

frame-0.

When frame-5 is captured, features 'red' and 'orange' come to an end as the flow tracking of these features fails, only feature 'green' is an existing feature that is updated, therefore we can not fix the frame in which 'green' is first observed in as the new information provides greater constraint on its position, the latest frame to not receive any updates to features within it is frame-2. The bundle adjustment window is therefore from frame-5 to frame-2 with the camera position ($[R|T]$) at frame-2 being fixed in space prior to the optimisation process along with all the features in frame-2 and their respective 3D positions. Fixing these parameters ensures that frames-(3-5) are optimised to the same scale as previous frames forcing a scale consistency across the rest of the SfM process. Many of the SfM approaches in Chapter 2 process data offline so are able to pick which image pair to initialise their reconstruction with and use as a basis for which to add more frames [26,31,37]. Other online approaches demonstrate initial camera motions that provide opportunity at the start of processing to select strong image pairs on which to build [24, 29]. In reality a vehicle just proceeds to drive without performing any set manoeuvres. This mandates that our SfM approach must start reconstruction from frame-0.

4.2.4.6 Pre-Bundle Adjustment Point Filtering

While BA can significantly and simultaneously improve the 3D point accuracy and camera poses, it is not immune to error and can be corrupted by a number of poor point-matches. A point filtering stage prior to the BA is required in order to provide it with sufficiently high quality point matches, to ensure reliable results. We conducted a series of tests to assess the impact of filtering the input points have on the final result of bundle adjustment by selecting an image sequence of 60 frames and examining the final mean reprojection error of the final bundle adjustment phase. The following rules are the filter strategies used:

- F1 - All points tracked with optical flow are sent to the BA stage, therefore it attempts to optimise camera poses so that every tracked point has its reprojection error minimised.

- F2 - Only tracked points with a reprojection error of $<$ 20px as used in the BA stage.

- F3 - Only tracked points with an initial reprojection error $<$ 20px OR if they have been previously bundle adjusted they must now have a reprojection error $<$ 5px.

The results presented in Table 4.1 show the reprojection error returned from the bundle adjuster [124]. The results clearly show that filtering is required before proceeding to the BA phase. In many cases, filter strategy 1 (F1) results in BA failing to converge on a solution, often resulting in a meaningless mean reprojection error of $> 10^2$px. The strategies of F2 and F3 appear to provide similar results when examining the output from BA only. However, when we investigate further the output reconstruction the conclusion changes. Figure 4.13 shows the reconstructed sparse point cloud after the full sequence has processed using each filtering strategy. Top to bottom, shows sequence 1 to 6 respectively. Left to right, shows filter strategies F1 to F3 respectively. We can see in Sequence 1 (top, left) that with no point filtering the resulting point cloud appears comparrable with the output from using filter strategies F2 and F3, however the post-BA reprojection error over the whole scene is large and considered a failure due to noisy outliers (Table 4.1). Sequences 2-6 using filter approach F1 (left) all show significant reconstruction failure, where F2 and F3 have once again resulted in near identical point clouds with similar reprojection errors. Sequence 5 demonstrates the need for the more strict point filtering strategy of F3. Here we show, despite similar reprojection errors for F2 and F3 (Table 4.1), the final point cloud generated with F3 (right) is significantly more dense than the one produced with F2 (middle).

	Seq. 1	Seq. 2	Seq. 3	Seq. 4	Seq. 5	Seq. 6
Image Seq.	8618	7335	1696	6293	6085	6485
Seq. Length	80	100	70	60	60	60
F1	Fail	Fail	Fail	Fail	Fail	Fail
F2	3.35	0.49	3.48	1.96	1.30	1.48
F3	1.51	0.55	3.40	2.04	0.65	1.57

Table 4.1: Mean post-BA reprojection error (px) of all features over variable length sequences using three different strategies for input point filtering prior to bundle adjustment.

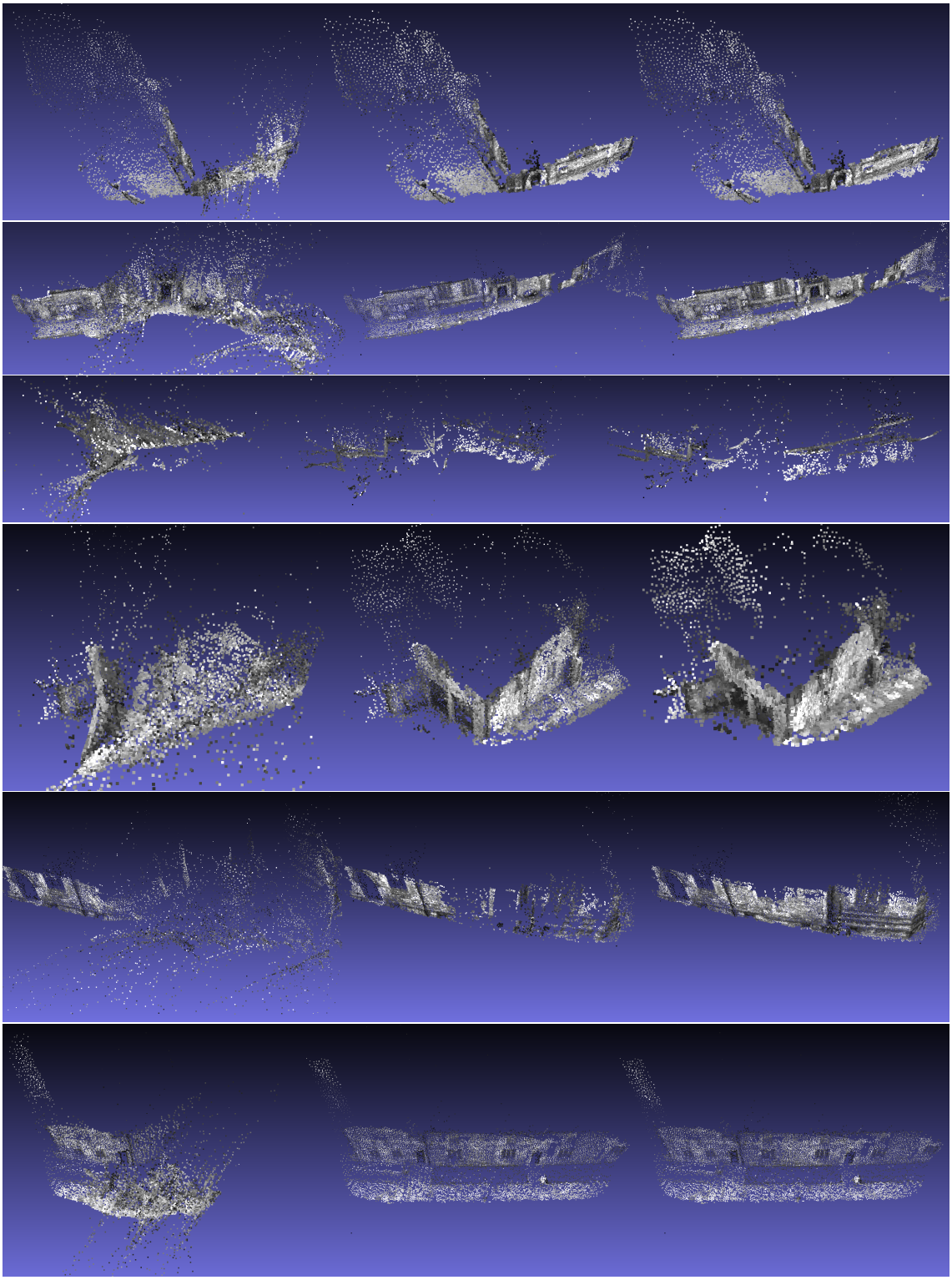

Figure 4.13: Sparse point clouds output from pre-BA filter tests. Top to bottom, sequence 1 to 6 respectively. Left to right, filter strategies F1 to F3 respectively.

4.2.4.7 Dynamic Bundle Adjustment Window Size

These next tests assess the performance of different bundle adjustment strategies in terms of which cameras to optimise over. It is possible to construct many strategies designed around different aspects, such as feature point weighting based on match quality [123] or temporal weighting [132]; here we investigate the impacts of a sliding temporal BA window size against our dynamic approach (Section 4.2.4.5). All cameras and points (that pass a given filter strategy) are considered for bundle adjusting up to a given window size. When a new image index exceeds the window size the last n-images are considered for bundle adjustment; where n in this case is the window size. We test fixed BA temporal window sizes of (10, 20, 30, 40, 50, and 60) frames and our dynamic window size approach using filter strategies F1 to F3.

Figure 4.14 shows the processing time of sequence 5 (Section 4.2.4.6) using a range of temporal window sizes over which to perform BA using different filter strategies. This shows that our dynamic window size selection is faster than most of the tested fixed window sizes and taking around the same time as a fixed window of the last 10 to 20 images. *(Note: A rapid fall in processing times to values* $\sim 10^{-2}s$ *indicates a failure of the BA process).* A fixed window of 10 appears to have comparable performance, over the whole image sequence, with our dynamic window size in terms of speed. However, by examining the quality, the median reprojection error of the points used within the BA window, (Figure 4.15) we can see the reprojection error using a fixed window size of 10 and 20 in filter mode F3 increases, indicating a failure. In this example, all of the BA window size approaches result in large reprojection errors for filter modes F1 and F2, even our dynamic approach has reduced performance. Using filter mode F3 with our dynamic BA window size achieves the best result in terms of maintaining low processing time and low reprojection error.

We show the dynamic window sizes that were automatically calculated in our approach (Figure 4.16) for sequence 5. This confirms that the similar results seen between our dynamic size and a fixed window of 10 and 20 images (Figure 4.14 and 4.15) would be expected, as in this case the dynamic window grows slowly with several intances being between 10 and 15 (Figure 4.16). Our dynamic window only

grows above 20 images towards the end of the sequence, therefore outperforms the fixed window size of 20 accross almost all of this sequence.

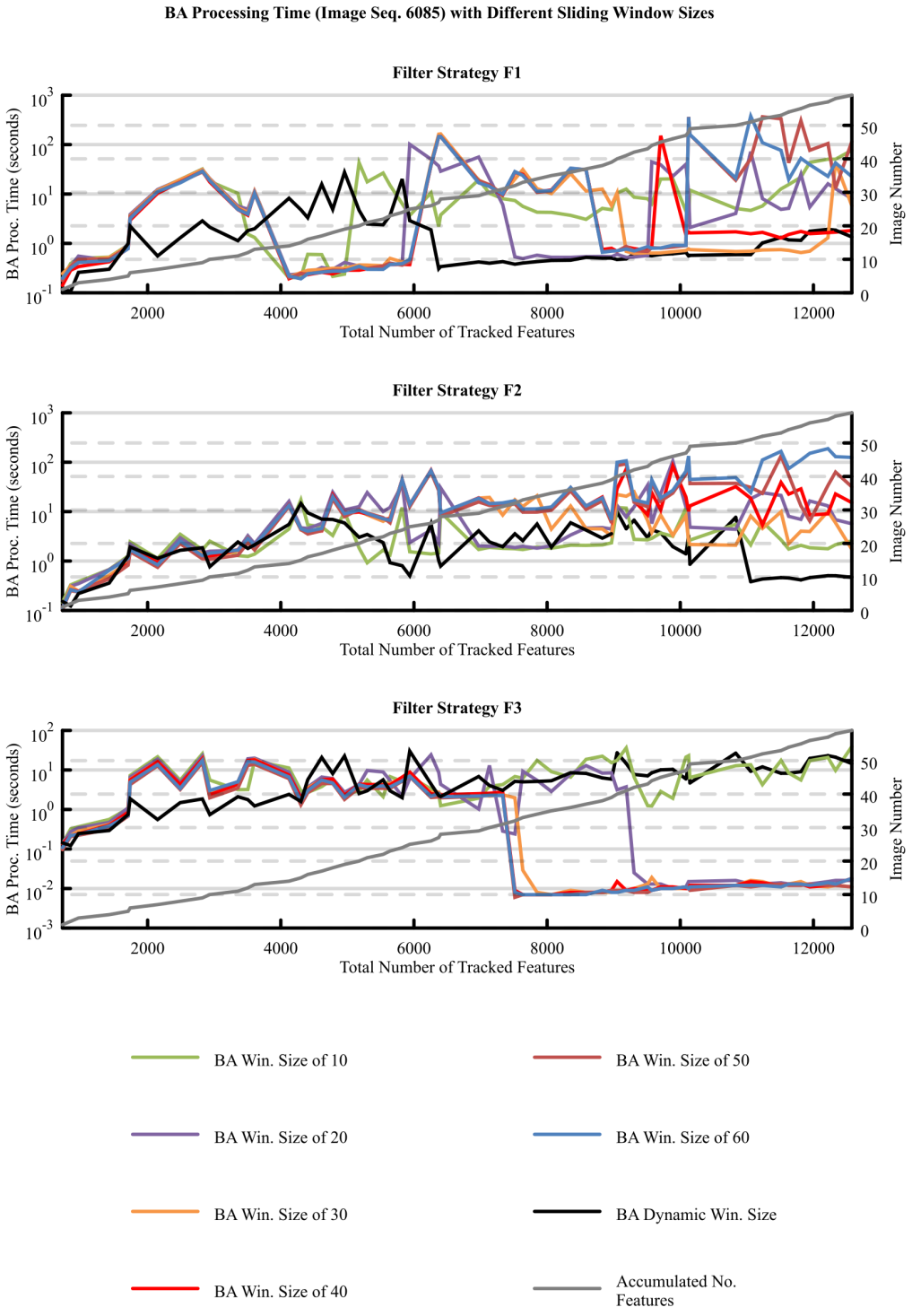

Figure 4.14: Bundle adjustment processing time as a function of number of features for different temporal window sizes and filtering strategies for sequence 5.

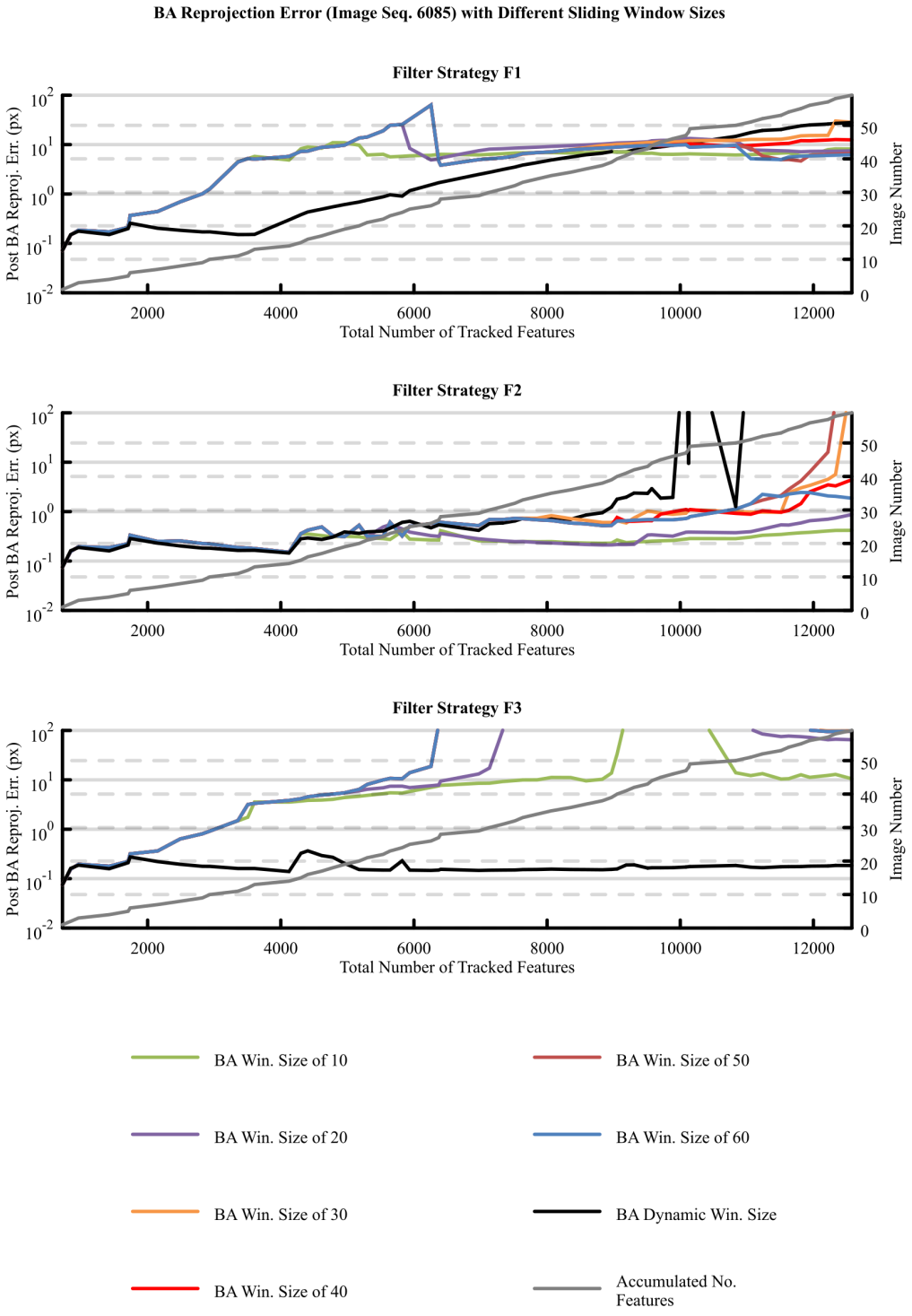

Figure 4.15: Reprojection error post BA stage of the SfM process for different sized temporal windows for sequence 5.

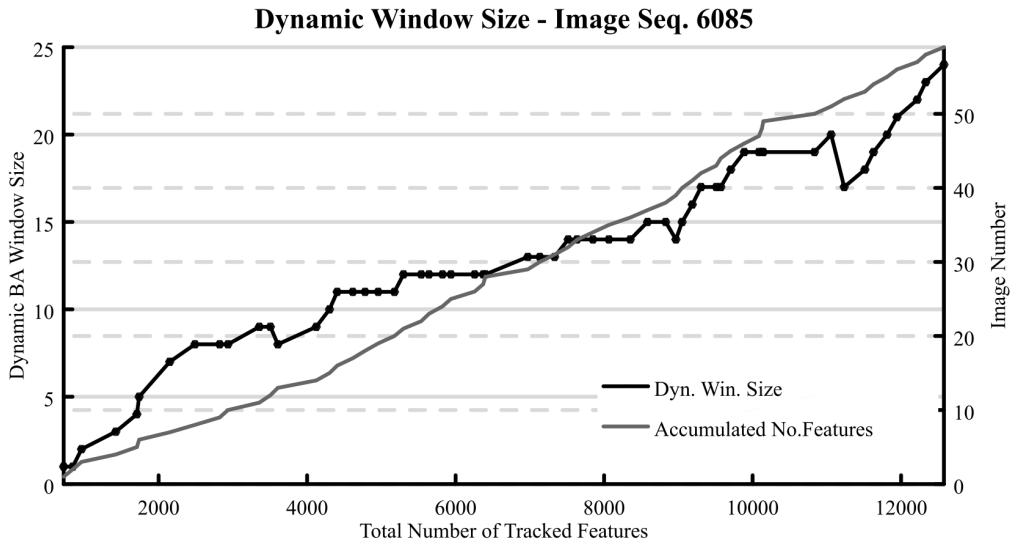

Figure 4.16: Dynamic window sizes used for our bundle adjustment approach of sequence 5.

Performing the bundle adjustment window analysis on a larger image set, sequence 2 (Section 4.2.4.6), shows a consistent picture. Figure 4.17 shows that using a dynamic window size achieves a lower processing time than most other window sizes and a similar processing time to that of the fixed window size of 10 and 20. The reprojection accuracy using a dynamic window also equals or outperforms the fixed window in almost all images in the sequence (Figure 4.18). In this sequence we can see the window size rarely goes above 20 images (Figure 4.19) and therefore it nearly always achieves a comparable processing time (Figure 4.17). Performing the experiment over all three filter modes demonstrates the improvement on the reprojection error that each filter approach has.

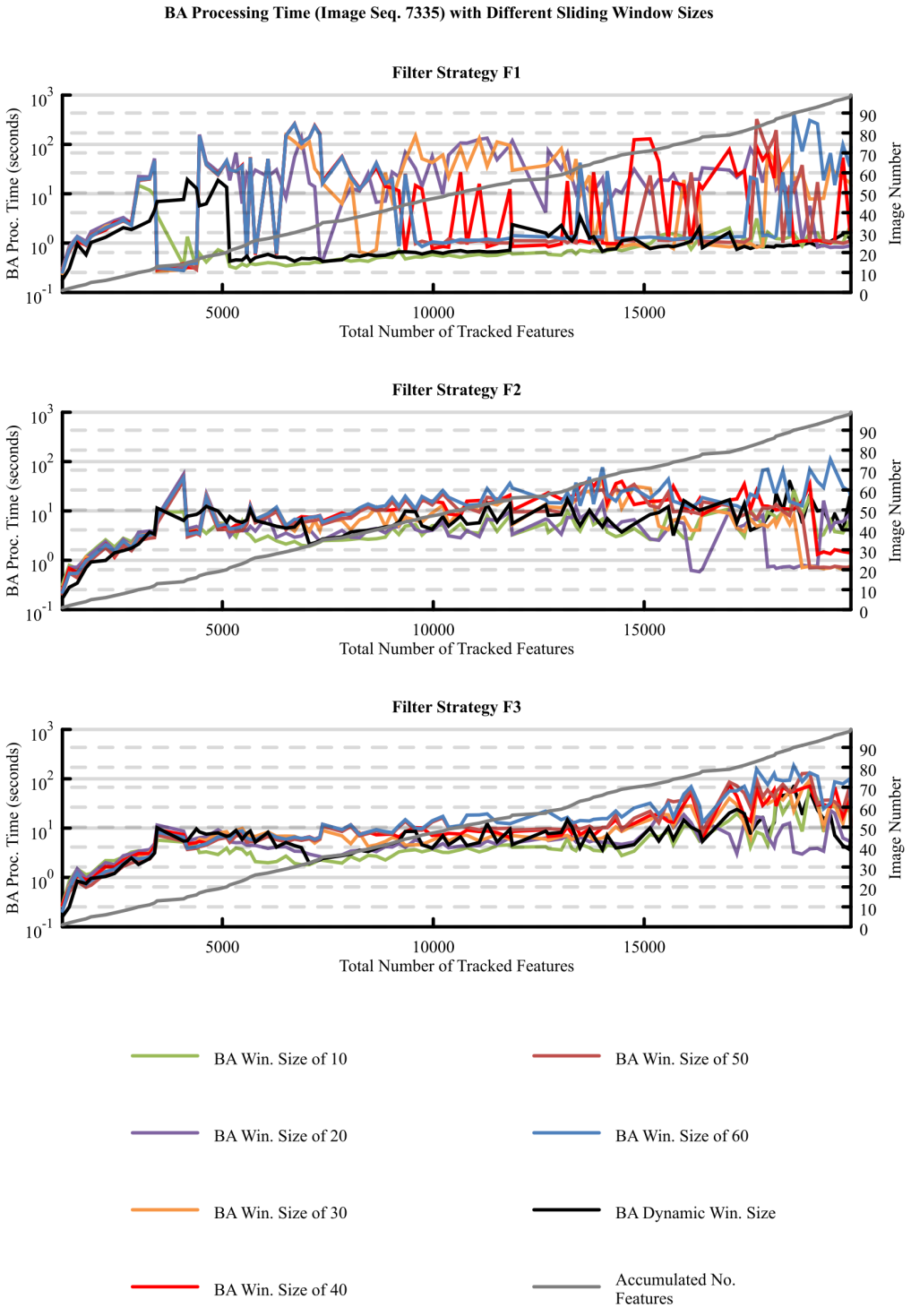

Figure 4.17: Bundle adjustment processing time as a function of number of features for different temporal window sizes for sequence 2.

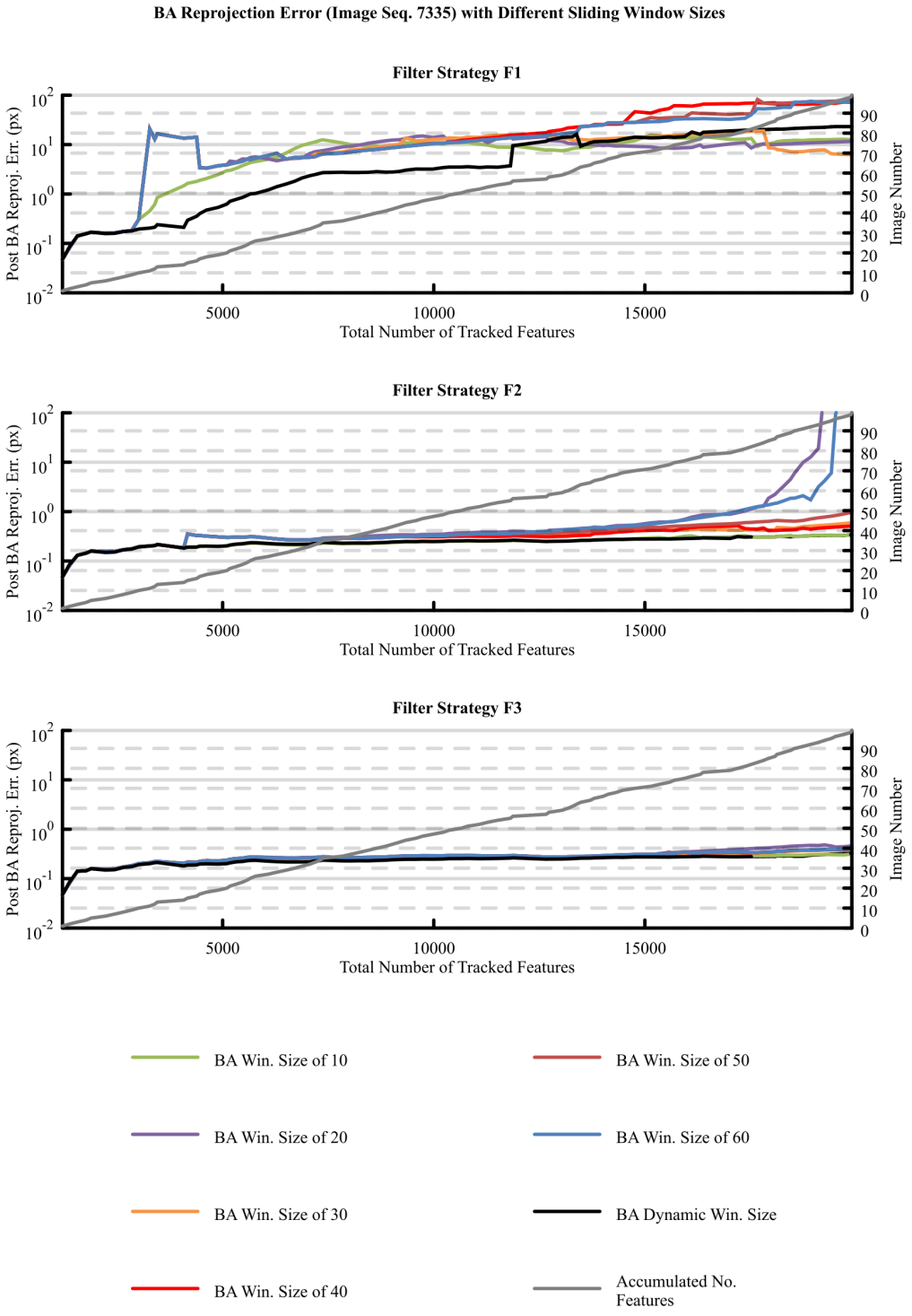

Figure 4.18: Reprojection error post BA stage of the SfM process for different sized temporal windows for sequence 2.

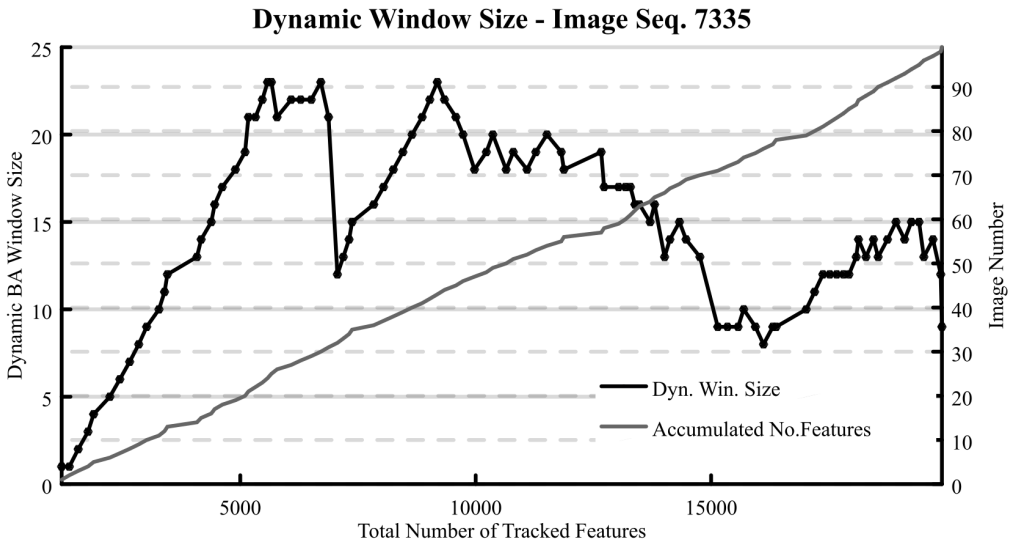

Figure 4.19: Dynamic window sizes used for our bundle adjustment approach of sequence 2.

With sequence 3 (Section 4.2.4.6) we see the same dramatic decrease in processing time (Figure 4.20) for all fixed BA window sizes using filter mode F2, indicating a failure case. Only our dynamic window is able to satisfactorily reconstruct the scene (Figure 4.13). From the processing time plot (Figure 4.20), filter mode F1 appears to be performing BA however the reprojection error (Figure 4.21) shows a large and increasing error indicating it can not successfully optimise for all tracked points. Only with filter strategy F3 do we see any results with a sufficiently low reprojection error for fixed BA window sizes. The dynamic window size used over this sequence, shown in Figure 4.22, illustrates that the optimal window size drops below 10 approximately halfway throught the sequence. At around the same point in the reprojection error plot (Figure 4.21) we see the fixed BA window sizes starting to increase in error and failing to recover.

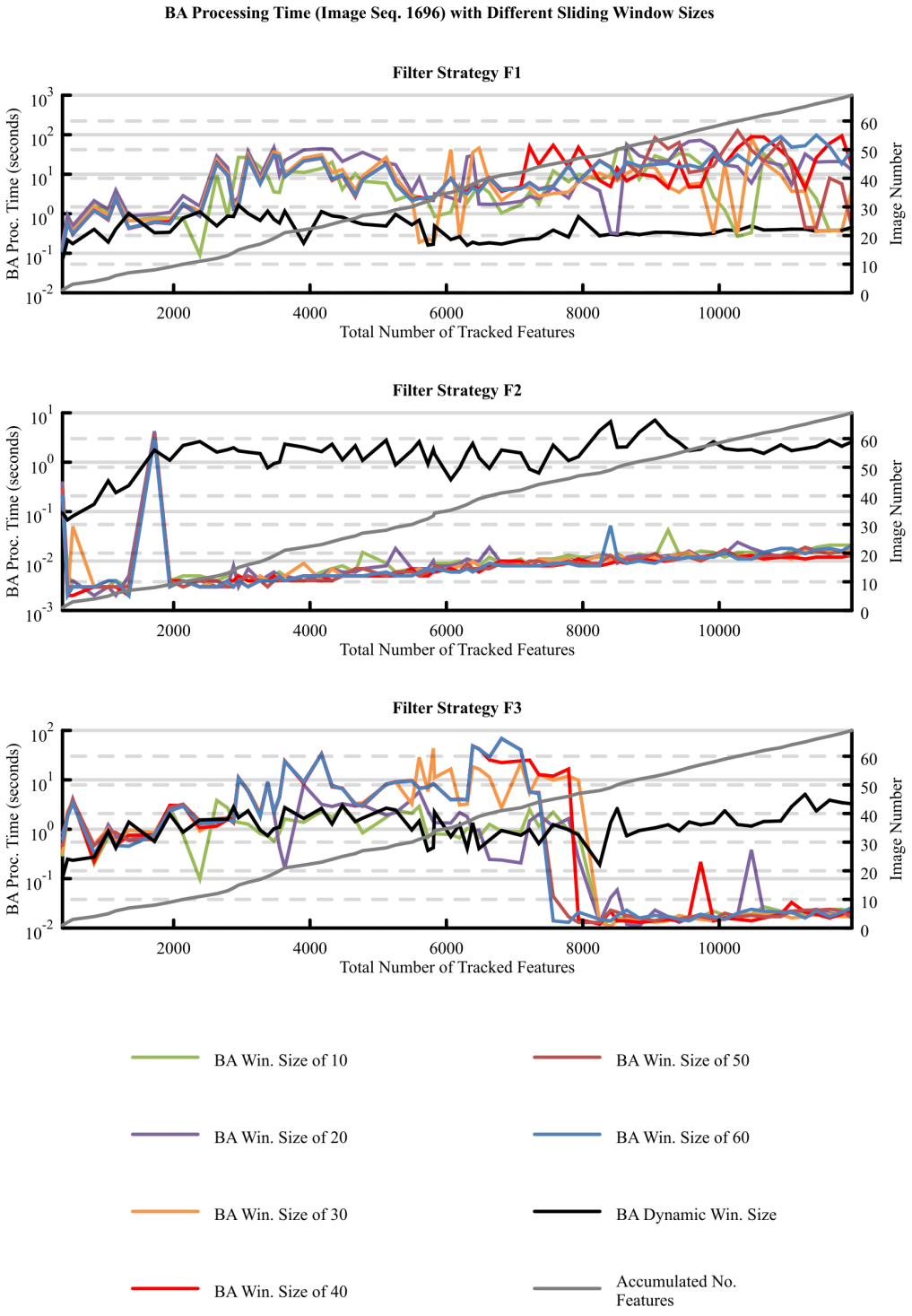

Figure 4.20: Bundle adjustment strategy test results from sequence 3 (Section 4.2.4.6).

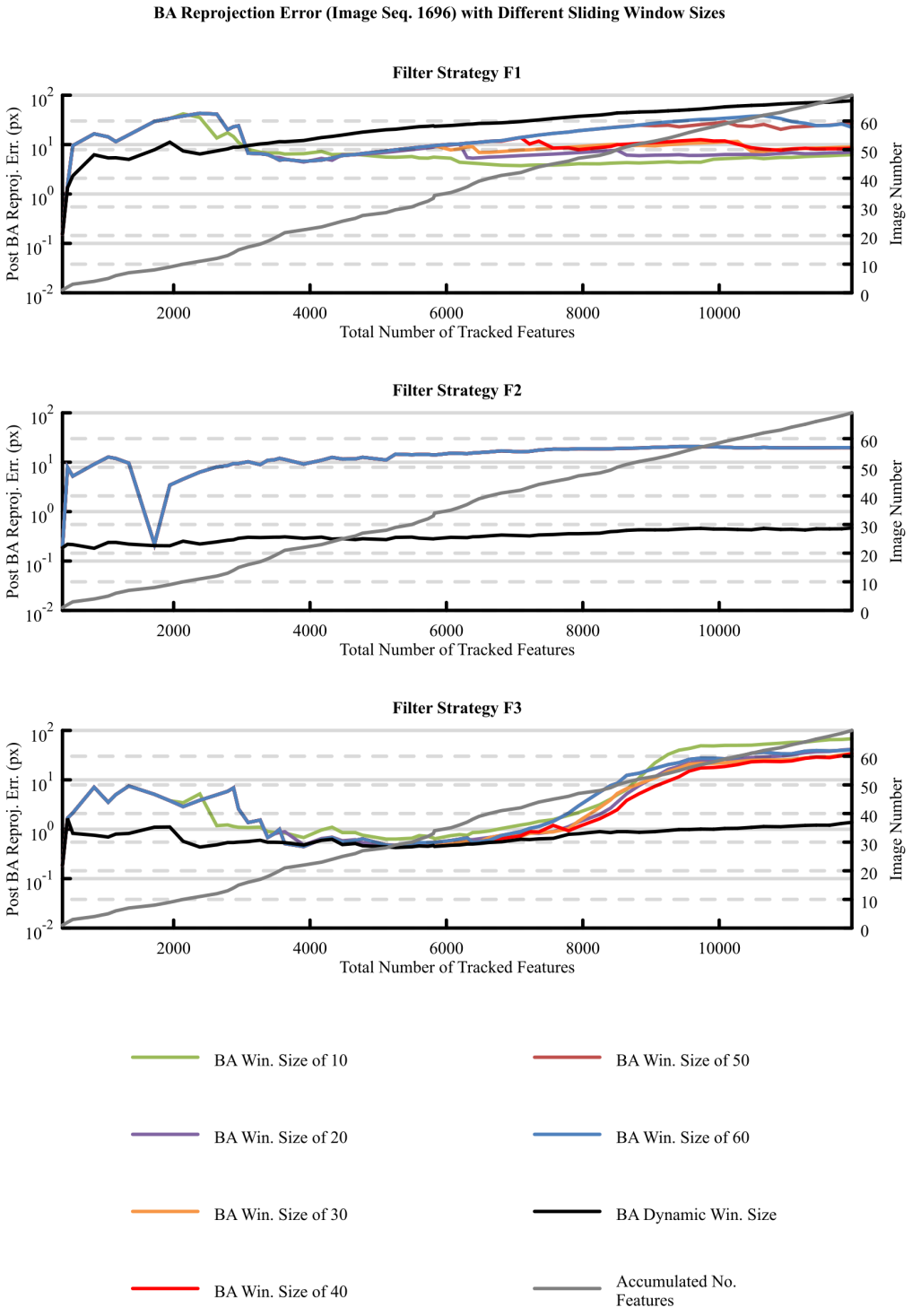

Figure 4.21: Reprojection error post BA stage of the SfM process for different sized temporal windows for sequence 3 (Section 4.2.4.6).

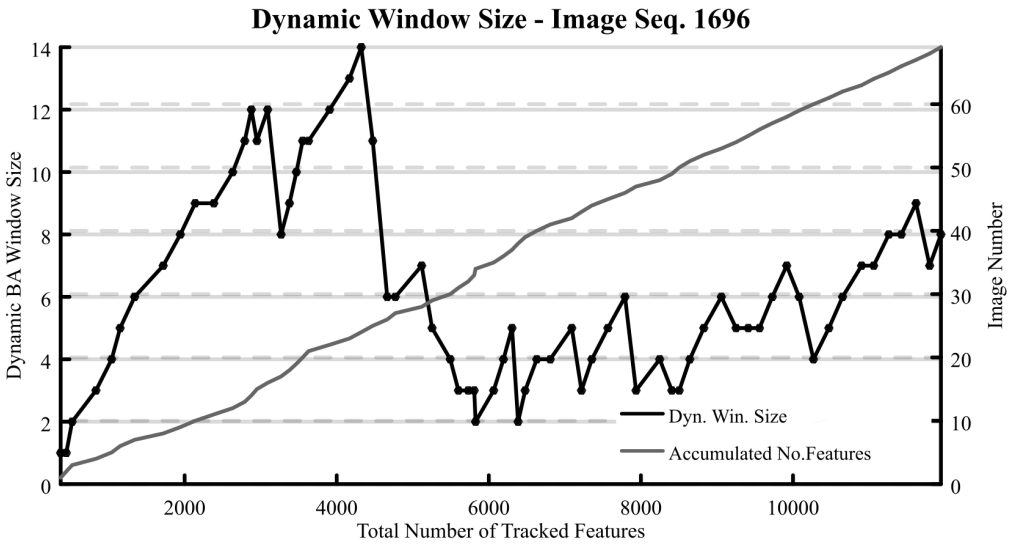

Figure 4.22: Dynamic window sizes used for our bundle adjustment approach of sequence 3.

Further plots for Sequence 1, 4 and 6 are in Appendix A.4.

This demonstrates that using an appropriate point filtering strategy (Section 4.2.4.6) and a dynamic temporal window for bundle adjustment can, together, make an effective base on which to build a structure from motion pipeline. We have also shown that although a fixed window can achieve good bundle adjustment performance it is susceptible to failure, requiring the data be reprocessed with a different window size, a difficult task should the reconstruction be an online process. The dynamic bundle adjustment window size has shown to be more robust and efficient than a fixed window. Using a sliding temporal window for BA to reduce accumulated drift error is a common technique [126,133–135]. The approach used is usually a fixed window where the length is tuned to achieve a desired accuracy or processing time. Work by [134] opts for window sizes between 6-9 while [126] tests fixed window sizes between 25-50. A simulation of BA window sizes over the last n-frames where $n = 1 : 5$ by [20] is performed by randomly removing image features to reduce the available data. Although [20] only performs tests using a small range of window sizes (1-5) they conclude that "*window size has little effect for strong data, but becomes increasingly important as the data becomes weaker*". Given that we use relatively weak features and matching, compared to more robust detectors and descriptors (e.g. SURF or SIFT), this would appear to agree with their findings. However,

as illustrated by our tests, a larger BA window does not always result in improved reprojection errors.

4.2.5 SfM Densification using Dense Stereo from Motion

After a sub-sequence of the main image feed, from the monocular side-facing camera, has completed a BA window and achieved a high accuracy of optimisation, two adjacent frames are selected and their pose information is fed into the same stereo rectification algorithm used by the forward facing fixed stereo cameras [62]. When calibrating stereo cameras, a chessboard pattern is typically used to calculate the relative poses of the left and right imagers, determine the intrinsic matrix and the lens distortion coefficients, all in real-world scaled units due to the known dimensions of the chessboard features. In the case of stereo rectification for our image pairs selected from the SfM sequence, we already have the intrinsic matrix and lens distortion coefficients from the prior calibration of the monocular cameras. The relative pose information is refined during the bundle adjustment phase leaving the final scale parameter, the baseline, to be provided by the SVO solution.

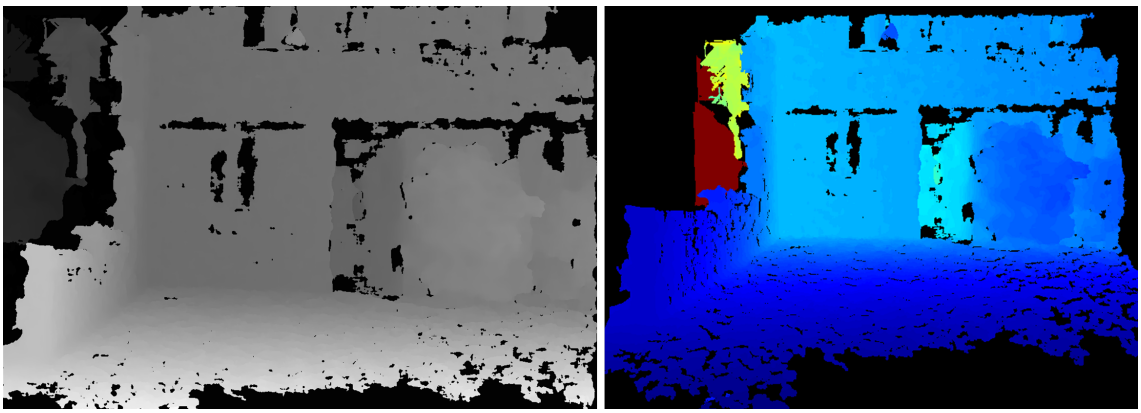

Figure 4.23: Left: Disparity map created from rectified image pairs using SGBM [5] dense stereo matching approach. Right: Colour-coded depth map (blue = nearest, red = farthest) viewed from a different, virtual, camera position.

Figure 4.2.5 left, is the resulting raw disparity map from an image pair after bundle-adjustment taken from the same sequence as Figure 4.2.4.5. In the right we see the point cloud generated from the disparity map, rendered with colour coding to illustrate depth. The process is repeated multiple times along the image sequence

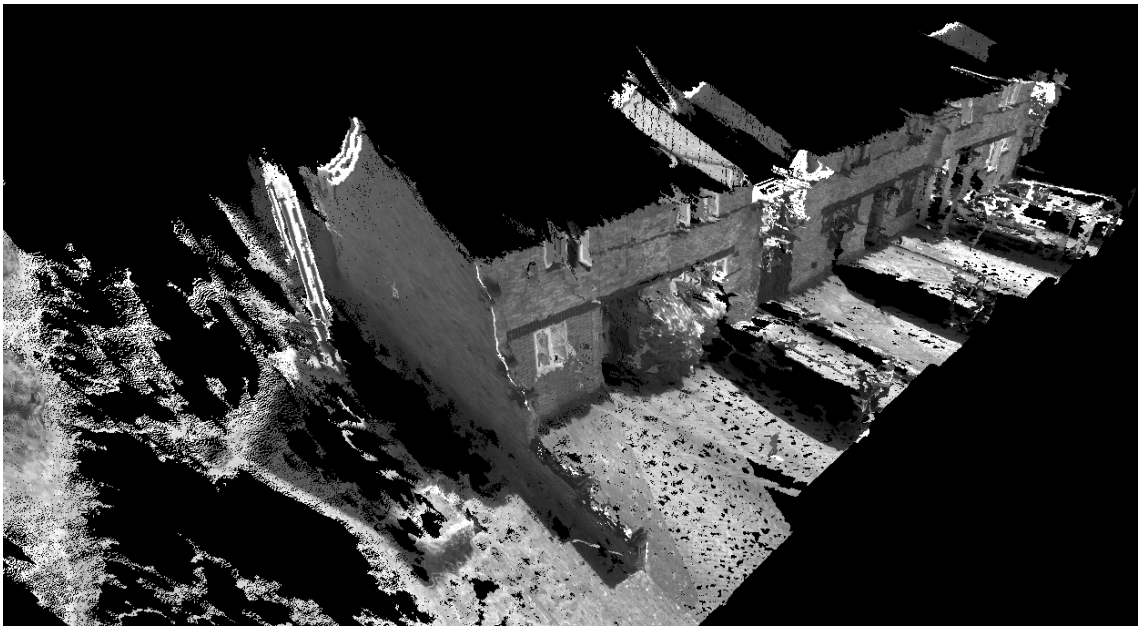

Figure 4.24: Dense raw point cloud reconstruction from SfM process using SGBM [5] for densification.

to reconstruct a full point cloud of the scene observed in the sequence, this is shown in Figure 4.2.5 as a greyscale point cloud where each point colour is set from their source pixels in the input image sequence. Comparing our result to that in Figure 1.1 shows that we have successfully reconstructed the street light to the left of the house and the low wall at the side of the drive. The importance of this stage is shown in Figure 4.25, where the point cloud attributed to the features only can be seen in both images. The addition of the dense processing demonstrates the vast amount of extra data compared to the sparse points.

Note: No point filtering or point cloud alignment has been applied to improve the output quality. This is the raw point cloud output from our dense SfM pipeline aligned using the pose information from the sparse bundle adjusted SfM process only, no point cloud alignment is performed.

The approach we use is optimised for planar camera motions; as the camera is mounted on a vehicle it is not likely to experience great amounts of rotation around its optical axis (vehicle pitch) or around the axis of motion (vehicle roll). It will however be subjected to rotations around their vertical axis (vehicle yaw), this has the effect of creating a 'toe-in' or 'toe-out' stereo pair. The rectification process can compensate for this to an extent, given a sufficiently fast image capture rate

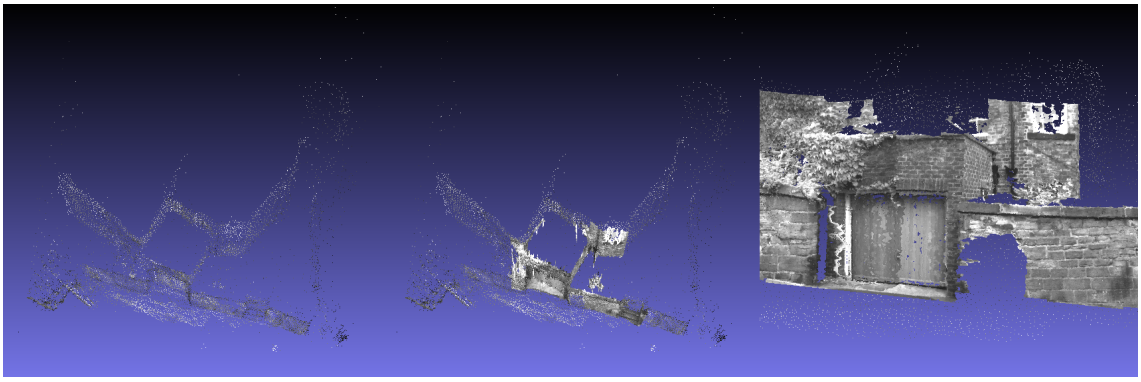

Figure 4.25: Reconstruction of a short image sequence viewed from a novel camera position high above the scene. Left, sparse points only. Middle, addition of dense reconstruction showing good structural agreement with sparse points. Right, middle point cloud viewed from a different viewpoint near ground level.

with respect to the vehicle speed, the camera-to-camera rotations are small and do not cause an issue when rectifying the chosen stereo pair. Figure 4.26 shows the reconstruction quality that can be obtained with this approach.

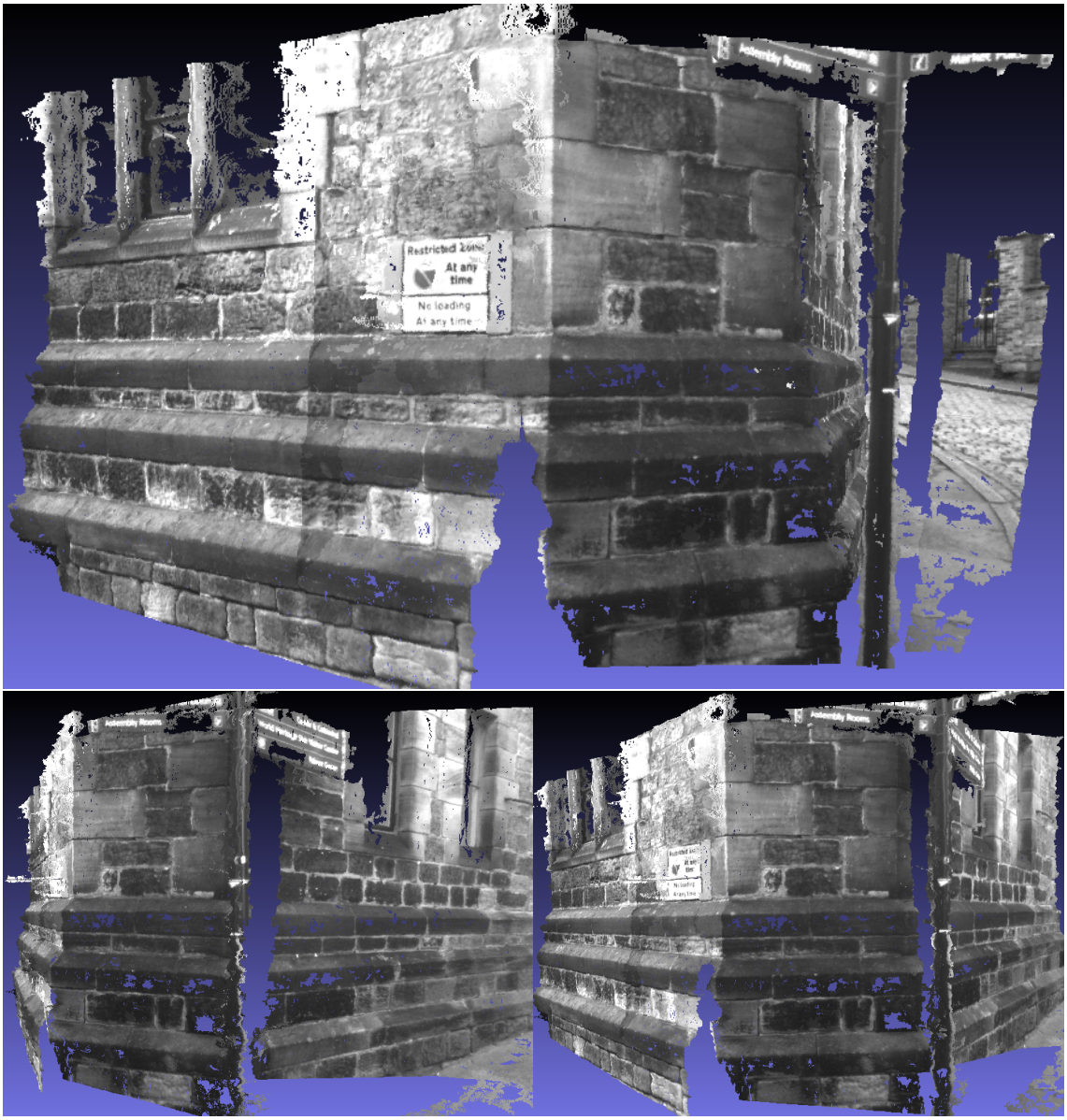

Figure 4.26: Three views of dense reconstruction around a corner from a left-hand turn.

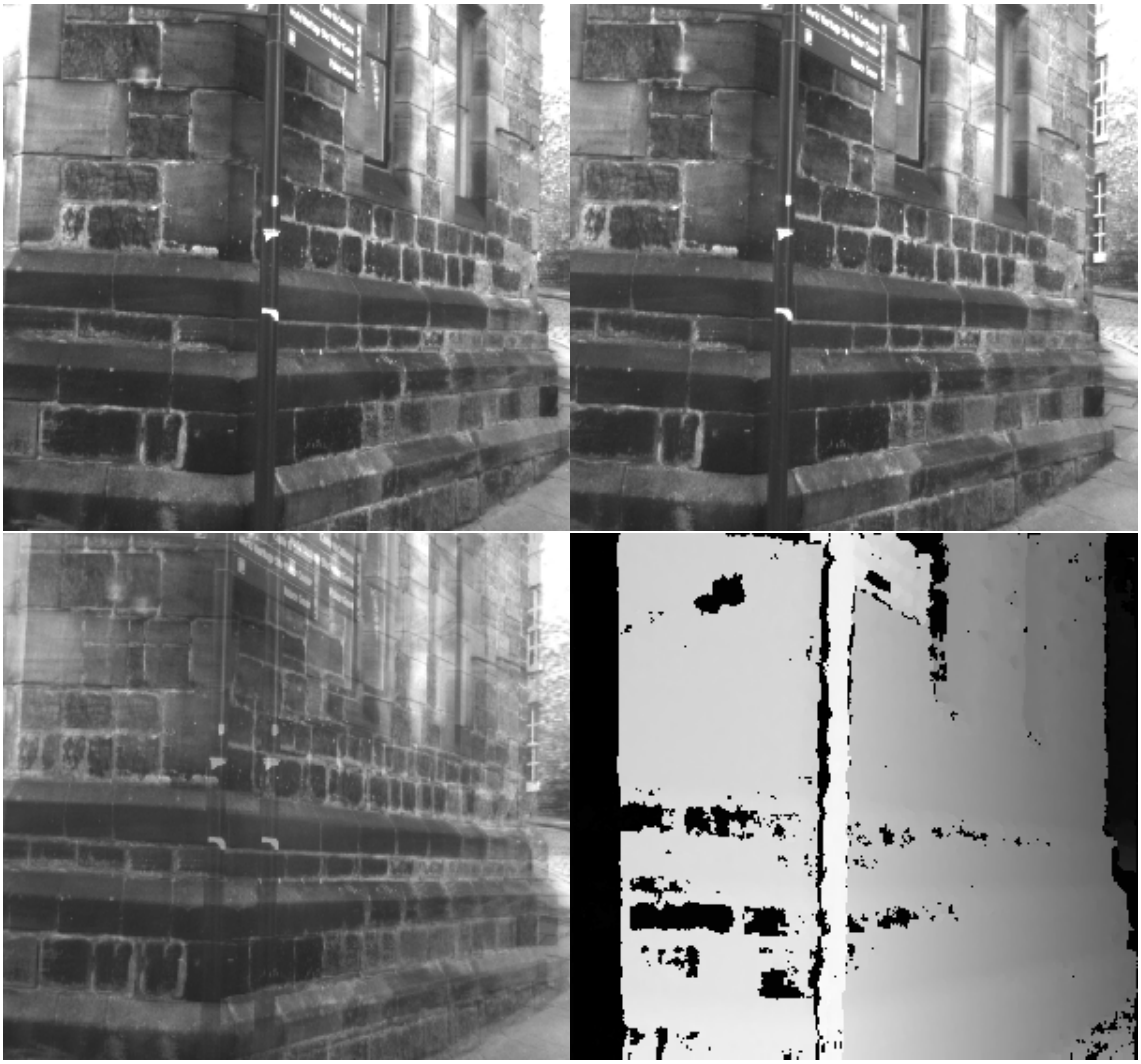

Figure 4.27: Example sequential images from single side facing camera and the densification process that produces almost pixel wise dense depth maps. Top left and right, are frames n and $n+1$ respectively post rectification. Bottom left is an overlay of each rectified image. Bottom right, shows the densification using SGBM [5]

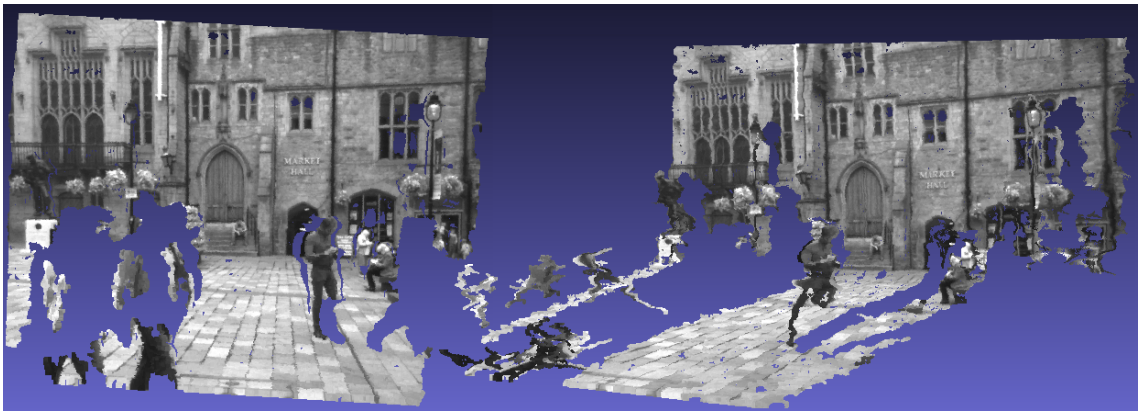

Figure 4.28: Dense reconstruction from our Durham, UK dataset viewed from two different virtual viewpoints to illustrate the 3D nature of the scene.

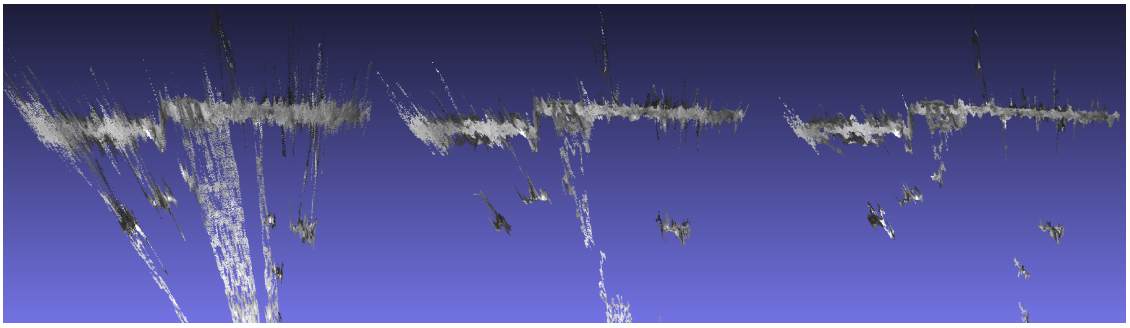

Figure 4.29: Top view of Figure 4.28 showing different baselines for SfM densification. Left to right, densification performed using frames n to $n+1$, n to $n+2$, and n to $n+3$ respectively.

Having the ability to select which pairs of monocular images to apply stereo rectification to, prior to dense stereo processing, allows improved triangulation accuracy on distant scenes. In Figure 4.28 we show the dense reconstruction of a part of Durham, UK. The range of the main building in the background is approximately 30m, the stereo matching error, manifesting as range error, can be clearly seen in Figure 4.29. Here we see, in the left image, the top-down view of the reconstruction created using two adjacent monocular frames (n and $n+1$) with baseline, B, and the associated 'streaks' of range error creating the fuzzy appearance of the wall. Using the same initial frame, n, but matching to $n+2$ creates a bigger baseline, $2B$, therefore reducing the range error, Equation 3.5 in Section 3.2.

Note: for illustration in this example we assume a constant vehicle speed, therefore the distance between each monocular image captured is therefore approximately

the same. At long ranges the inter-frame distance is going to be significantly less than the scene range, Z, so we assume this is constant. As the camera is also the same for each image the focal length, f, is constant and using the same stereo algorithm Δd also remains the same. Equation 4.4 rearranges to Equation 4.5 showing the inverse relation between baseline and range error.

$$\Delta Z_1 = -\frac{Z^2}{fB_1}\Delta d \qquad \Delta Z_2 = -\frac{Z^2}{fB_2}\Delta d \tag{4.4}$$

$$\frac{\Delta Z_1}{\Delta Z_2} = \frac{B_2}{B_1} \tag{4.5}$$

The right image, in Figure 4.29, being created using frames n to $n+3$ or a baseline of $3B$, therefore has range error on 3D triangulation of 1/3 of that seen in the left image.

4.3 Summary

We have shown using simple features uniformly distributed and matched across frame sequences using optical flow we can create an SfM approach that enables semi-dense reconstruction using feature points and an online densification process using dense stereo techniques. The use of optical flow and the nature of its one-to-one matching removes the need to search large feature spaces for historic matches. This allows for simple rearranging of 3D-2D correspondence lists to augment the bundle-adjusted point cloud data with new 2D feature points as they arrive from the image sequence. Our dynamic bundle-adjustment window of the last n-frames enables camera poses to be refined in a fast and efficient manner and allows the use of stereo rectification and dense stereo processing to produce near-pixel-wise density of depth information. We demonstrate a 3D point reconstruction rate that exceeds prior work in this area of large-scale high-resolution outdoor constructions (results are presented in Chapter 6).

Chapter 5

Moving Object Removal

In this chapter we exploit prior data available from the dense stereo processing and SVO stages (Chapter 3) of our reconstruction pipeline to perform a novel approach to generic dynamic object removal.

5.1 Introduction

Image driven approaches to 3D scene mapping suffer from an inherent problem of dynamic object separation from the otherwise static background. RADAR mapping [136] has the advantage of being able to discriminate dynamic objects based on spectral returns caused by the Doppler effect, however it typically lacks the angular and spatial resolution needed for mapping on the cm scale required in automotive environments. LIDAR [12] has good spatial resolution in the axis of the beam, (i.e. range); however, it often has poor azimuth angular resolution, usually with even poorer elevation angular resolution, resulting in sparse scene coverage at longer ranges. The instantaneous point sampling of LIDAR also succumbs to the same limitations in dynamic object detection as the stereo case considered here. As discussed in Chapter 2, spatiotemporal reconstruction methods, specifically SfM, rely on the assumption that the majority of tracked features are attributed to the static background. They typically naturally reject dynamic scene components at the RANSAC processing stage. Identification of dynamic object from stereo images is more difficult as there is no temporal offset between the left and right images.

5.2 Process Overview

As discussed in Section 2.5 the existing approaches use processing techniques that are typically not directly applicable to 3D mapping. Our approach aims to greatly simplify the solution to this problem by leveraging the wealth of recoverable information already available from a stereo camera in motion (e.g. depth, odometry, optic flow, structure-from-motion etc.) that is also required for the mapping process. We propose a pixel-wise approach to tackle the issue of dynamic object removal from 3D maps in the general sense whilst imposing limited additional computational load. By using intermediary data from the odometry driven stereo mapping process we can highlight the dynamic objects in the scene so as to remove them prior to the final mapping stage. As outlined in Figure 5.1, we first compute the dense stereo disparity map for the initial stereo image pair. We then process the subsequent captured frames to again produce a dense disparity map. Furthermore we also process the odometry of the camera between stereo pairs at t and $t+1$ to obtain the platform motion using stereo visual odometry, SVO [63]. Both dense stereo and SVO are already required for the 3D mapping solution, hence at this stage no extra processing of the raw input images is required. Using the calculated platform position, from SVO, we can now reproject disparity maps into a common virtual view point. From this disparity reprojection calculation we can synthesise an Optical Flow From Disparity (OFFD) map. The dense optical flow map is then used to remap the raw intensity image to the same virtual view point. This process of remapping disparity and intensity images to simulate observation from a different viewpoint allows us to align them more accurately than a traditional affine transformation. A 2D projective transform, as eluded to by the image space driven techniques of [98,99,137], does not take into account the 3D nature of the scene. By contrast, our approach uses full scene structure and camera motion information to reproject a 2D image with 3D constraints.

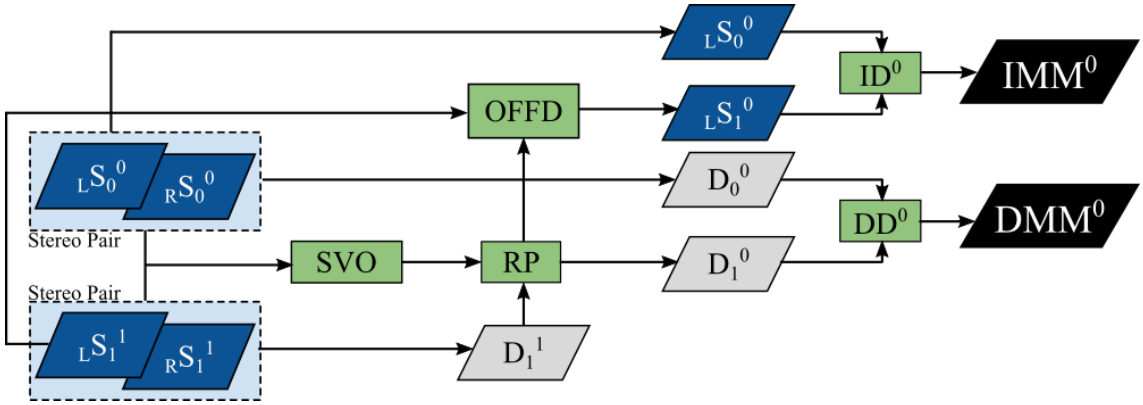

Figure 5.1: Main processing overview for moving object detection from a moving stereo camera platform. The naming convention in the diagram is as follows, ${}_{H}M_{T}^{V}$ – H is the handedness of the frame (Left or Right), M is the image or matrix, T is the time it originates from, V is the viewing time it is projected into. Hence, ${}_{L}S_{1}^{0}$ is the left stereo intensity image from $t = 1$ remapped into $t = 0$. S - Stereo Intensity Image; D - Disparity Map; OFFD - Optical Flow From Disparity; DMM - Disparity Moving Mask; IMM - Intensity Moving Mask; SVO - Stereo Visual Odometry; RP - Reprojection Calculation; DD - Disparity Difference; ID - Intensity Difference.

Based on the stereo calibration approach of [19], we recover stereo disparity and hence scene depth based on the approach of [5]. For clarity, with knowledge of the stereo cameras configuration we can construct a matrix, Q (Equation 5.1) which allows for projection of a 2D disparity image point into 3D world-scaled points (Equation 5.2).

$$Q = \begin{bmatrix} 1 & 0 & 0 & -c_x \\ 0 & 1 & 0 & -c_y \\ 0 & 0 & 0 & f \\ 0 & 0 & a & b \end{bmatrix} \tag{5.1}$$

$$\begin{bmatrix} X \\ Y \\ Z \\ W \end{bmatrix} = Q \begin{bmatrix} x \\ y \\ d \\ 1 \end{bmatrix} \tag{5.2}$$

Where X, Y, Z are 3D coordinates scaled by W, (x,y) are disparity image pixel locations with disparity value d, $a = \frac{-1}{B}$, $b = \frac{c_x - c'_x}{B}$, B is stereo camera baseline in meters, f is camera focal length in pixels, $c_{(x,y)}$ is principal point of left camera and

$c'_{(x,y)}$ the right. The application of Equation 5.2 to all recovered disparity values results in a 3D point cloud as per common formulation [19].

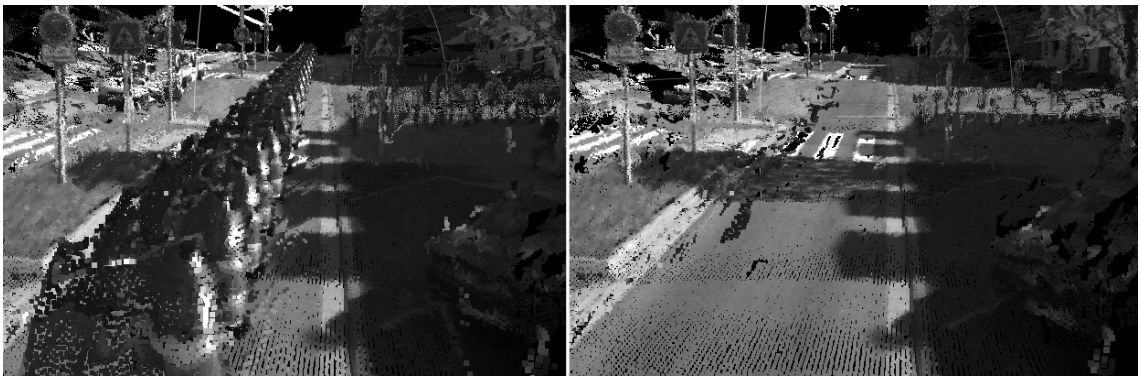

Figure 5.2: Left: A car driving forwards, ahead of the logging platform, reconstructed multiple times cluttering up the global point cloud. Right: Moving object removed from the final reconstruction.

5.2.1 Visual Odometry

We use the stereo visual odometry approach of [63]. The basis of this approach is to match 2D image features between stereo image pairs at time t and $t+1$ and to triangulate them using knowledge of the stereo baseline to recover real-world scaled 3D camera poses in the form of a rotation matrix, R, and translational component, T. These triangulated features from time t are then matched to features in $t+1$. The camera motion from t to $t+1$ can now be calculated by minimizing the reprojection error of the now known 3D points from t into that of $t+1$. Although SVO has been chosen as it provides the odometry in real-world units, a monocular camera technique performed in [59] has been demonstrated to outperform [63] in some cases. The work of [59] does however, require a region of the road in front of the vehicle being visible and the camera height above the road being known. By contrast, our chosen SVO approach is without such constraint or assumption. Our dynamic object removal approach is essentially independent of the odometry source (optical, mechanical, IMU or GPS) provided it has sufficient accuracy comparable to that found in [63].

5.2.2 Scene Mapping

To build a 3D point cloud representation of a scene we combine the outputs from dense stereo vision (Chapter 3) and stereo visual odometry (Section 5.2.1). From the SVO stage we know the camera pose at every frame with dense stereo vision providing the 3D scene structure in the form of a point cloud, as shown in Figure 5.2 where we also see the prevalence of the dynamic object problem in such cases.

5.3 Dynamic Object Removal

Central to our approach is the fact that a moving point in space is defined as a rate of change of position in $[x, y, z]$. Each stereo image pair is taken at discrete time intervals, hence this frame-to-frame capture interval is our minimal detectable rate of change. In order to detect and isolate object motion within the scene we must hence match inter-frame 3D positions on a point-wise basis, between the spatially adjacent stereo camera positions obtained via SVO, within reasonable computational bounds. To enable a point-wise image comparison we perform a scene structure aware projective transform of both disparity and intensity images of consecutive frames into a common camera position.

5.3.1 Disparity Projections

Disparity at each stereo pair varies with object depth, motion and relative camera motion within the scene, preventing direct disparity map comparison between consecutive frames. To compensate for the camera motion we transform the triangulated point cloud by the inverse of the camera motion, then project the new motion-compensated 3D points into a synthetic disparity image, Equation 5.3. Subsequently, we update the disparity map values to reflect their new distance from the virtual viewpoint, Equation 5.4.

$$\begin{bmatrix} u \\ v \\ 1 \\ 0 \end{bmatrix} = CMQ \begin{bmatrix} x \\ y \\ d_0 \\ 1 \end{bmatrix} \tag{5.3}$$

$$\begin{bmatrix} r_{31} & r_{32} & r_{33} & t_Z \end{bmatrix} \begin{bmatrix} x - c_x \\ y - c_y \\ f \\ \frac{d_0}{B} \end{bmatrix} = \frac{d_0 f}{d_1} \tag{5.4}$$

Where $M = [R|T]^{-1}$, R, T are the rotation and translation components of the camera motion from SVO respectively; C is camera intrinsic matrix; $c_{(x,y)}$ is principal point of the camera; $(x,\ y)$ are 2D image coordinates in source disparity image; $(u,\ v)$ are 2D image coordinates in synthetic disparity image; r_{rc} are the rotation components from M at row r and column c; f is camera focal length in pixels; (d_0,d_1) are source and destination disparity respectively; t_Z is the translation component of platform motion in Z-axis.

The transform of Equation 5.3 creates a new synthetic disparity image that corresponds to a virtual camera at the location of the previous camera position. A scaling transform, Equation 5.4, updates disparity values in the synthetic disparity map to reflect the new distance points lie away from the virtual camera position. Figure 5.3, shows three disparity images from two stereo pairs. Top disparity image is at time t and middle disparity image at $t+1$, the bottom image is disparity at $t+1$ projected, via Equations 5.3 and 5.4, into the virtual camera position of t. Observing the red vertical lines shows how features such as windows, signs and backs of cars are now aligned, allowing for direct point-wise comparison of disparity at t and $t+1$. There are two ways this method can be used: (a) projecting forwards in time $t \rightarrow t+1$ or (b) back projecting in time $t+1 \rightarrow t$. In the case of forward camera motion it is preferable to use projection scheme (b). Back projecting into a previous frame maximises the usable data as there is a greater chance the image at $t+1$ lies entirely within the spatial bounds of the image at t.

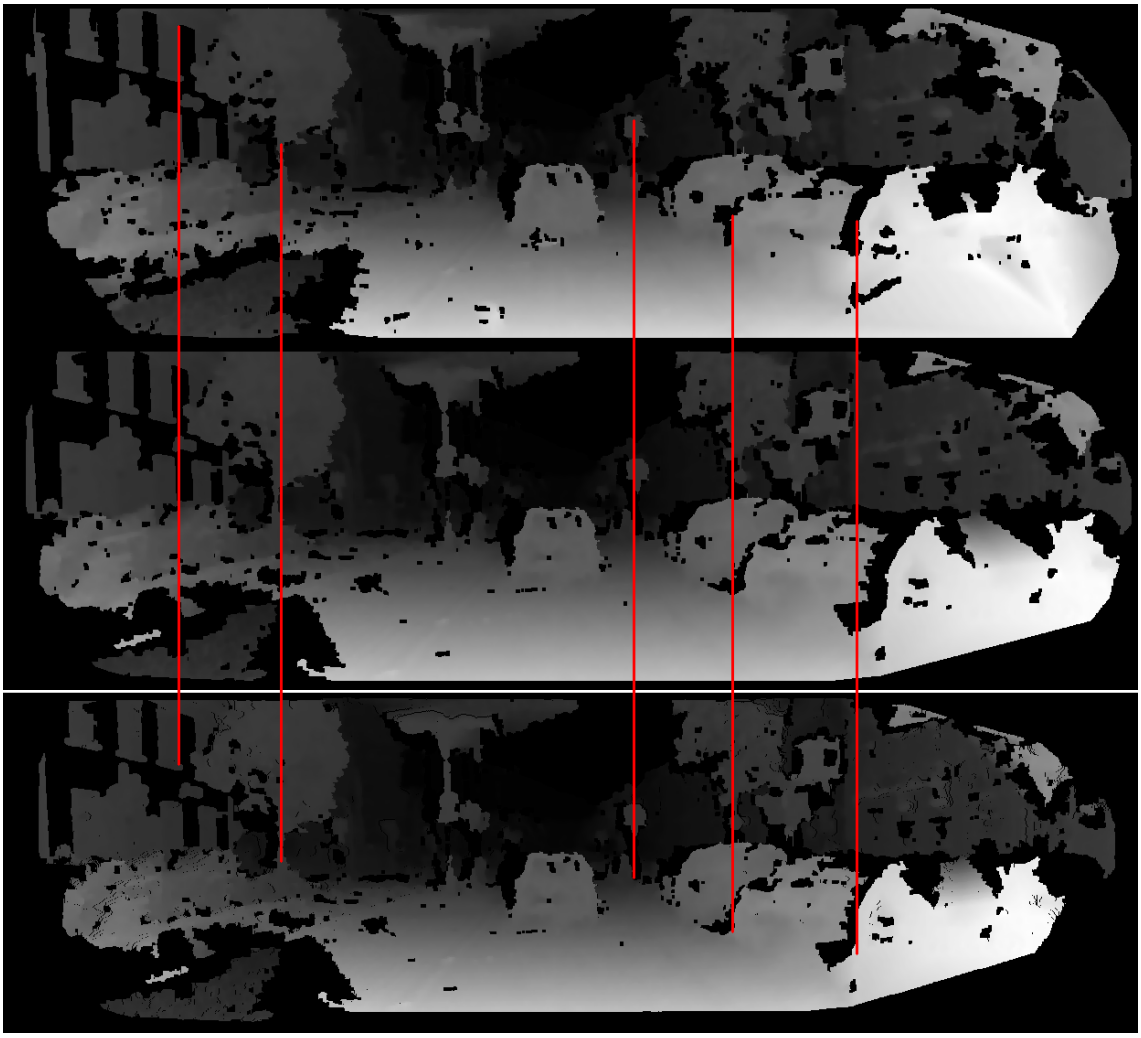

Figure 5.3: Top: Disparity map at t. Middle: Disparity map at $t+1$. Bottom: Disparity map at t+1 back projected into t. Vertical red lines illustrate the alignment between scene features at t, $t+1$ and $t+1$ back projected to t.

Spatial point-wise alignment of temporally separated disparity maps permits us to compute a binary moving object mask. This is done by performing a point-wise difference image between the two projection aligned disparity maps, creating a Disparity Difference map (DD, Figure 5.1). In Chapter 3 we demonstrated that the non-linear 3D triangulation error from various dense stereo matching algorithms can be represented by a disparity matching error in pixel terms in disparity space. The disparity maps we produce are calculated using using Semi-Global Block Matching (SGBM) [5]. The estimated SGBM stereo matching error, for real world data, is approximately e=0.1 pixels [102]. We use the higher threshold of 0.2 pixels to reduce threshold noise and allow for some positional error attributed to SVO solution.

Using this estimated accuracy metric we threshold the Disparity Difference map to populate the binary Disparity Moving Mask (DMM, Figure 5.1) (Equation 5.5).

$$DMM_{xy} = \begin{cases} 0 & DD_{xy} \geq e \\ 1 & otherwise \end{cases} \tag{5.5}$$

The DMM is used to reject regions of the disparity map used for the 3D reconstruction. Figure 5.2 illustrates the aggregated point cloud reconstruction before and after the moving object removal stage. However, Figure 5.4 illustrates a situation where disparity projections do not provide a robust solution for object removal. This occurs when an object is moving approximately perpendicular to the path of the camera and is large enough or moving slowly enough such that depth variation is always below error threshold, e. The DMM in Figure 5.4 shows a large depth change in front and behind the moving object, however the region covering the object centre is tagged as valid and will not be removed from the final map. To detect this class of motion an additional step must be taken.

Figure 5.4: Left: Intensity image from left stereo camera. Right: Threshold of disparity difference image or DMM.

5.3.2 Optical Flow From Disparity

The failure mode demonstrated in Figure 5.4 can be mitigated by examining the intensity consistency between consecutive frames. Likewise with disparity map projections we reduce the search space by aligning the images so that direct point-

wise comparisons can be performed. Traditionally frame alignment would be done by applying a 2D homography transform, essentially performing image stabilisation [98, 99]. This approach is insufficient because homography transforms are 2D projective transforms that do not take into account the full 3D nature of the scene, so precise full image alignment is not possible from a single transform. An attempt by [137] to tackle this problem, for a monocular camera configuration, is to split the images into cells (approximately 16×16 px) that are each motion compensated and have a unique affine transform applied. The method presented here is to project every intensity pixel (i.e. analogous to a cell size of 1×1 px) by their optical flow response. Accurate dense optical flow techniques, such as [138], required for this level of intensity projection are computationally expensive. In contrast to our approach, using disparity projections, we avoid dense optical flow calculations entirely as scene structure aware remapping of 2D points at time t into a 2D point at $t+1$ has already been calculated in Equation 5.3. Figure 5.5 shows the full scene dense flow field for the failure case outlined above. Using the Optical Flow From Disparity (OFFD, Figure 5.1) we can remap the original intensity image from $t+1$ to t, Figure 5.6. Comparing the top image to the bottom image, using the vertical red lines as a guide, we can see that the static scene components line up correctly regardless of the scene depth, whereas the dynamic pedestrians are clearly in a different position. Performing a point-wise comparison of aligned intensity images results in an intensity difference map (ID, Figure 5.1); applying an appropriate threshold yields an Intensity Moving Mask (IMM, Figure 5.1). Unlike the threshold used for the disparity moving mask, which is primarily dictated by the performance of the dense stereo algorithm used, the threshold applied to the intensity difference map is empirically chosen depending on the noise level of the images acquired from the cameras. Applying the IMM to our dense disparity maps we mask out the regions pertaining to dynamic objects as shown in Figure 5.7.

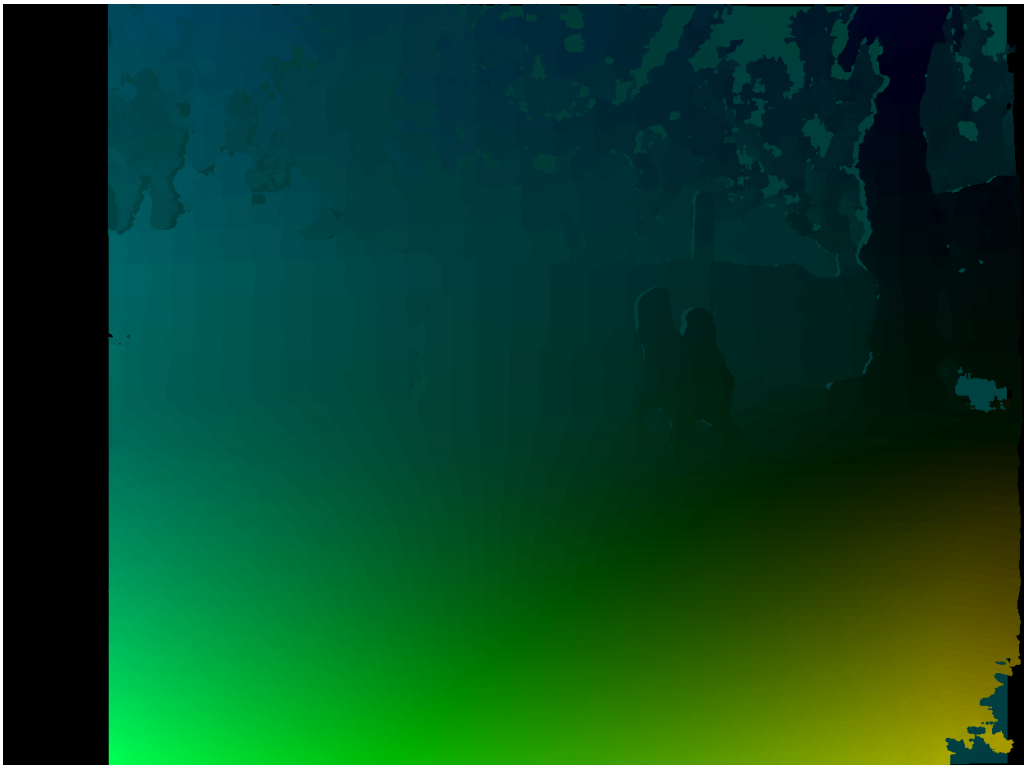

Figure 5.5: Flow from disparity of the illustrated failure mode in Figure 5.4 with the vehicle driving forwards and right (flow direction and magnitude are represented respectively by the hue and value channels of the HSV colour space).

Figure 5.6: Top: Intensity image at time t. Middle: Intensity image at $t+1$. Bottom: Intensity image at $t+1$ projected to t. Vertical red lines illustrate alignment of various features, demonstrating the reprojection accuracy. The white region on the left of the bottom image, this is unmatched region of the disparity map corresponding to the maximal disparity search window.

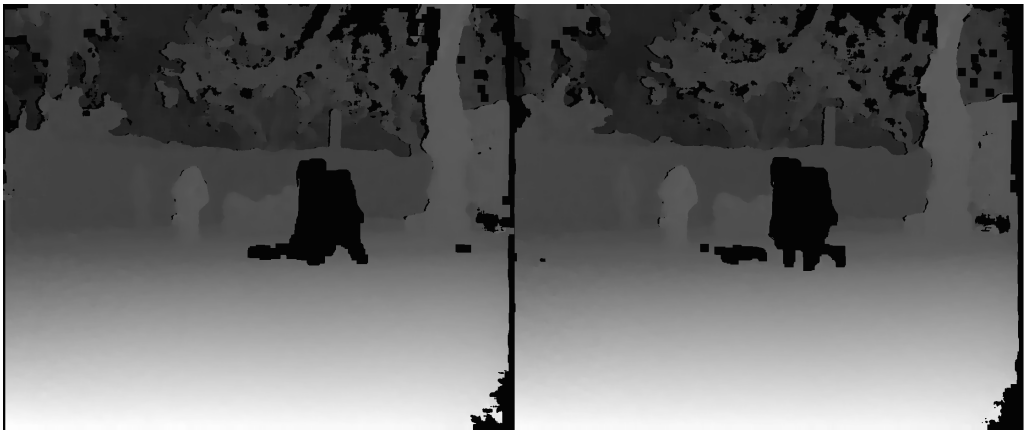

Figure 5.7: Disparity maps with moving objects masked out. Shadows cast by the moving objects are also masked out due to intensity image variation.

5.4 Results

The data sources used for this work are the popular KITTI stereo dataset [34] and our own image sequences. Each stereo camera has a horizontal field-of-view of approximately 45°, stereo image synchronisation is controlled by an Arduino microcontroller. Image data is collected in the form of greyscale images of resolution 1280×960px at a rate of 7.5fps. The vehicle speed for our dataset was no greater than 30mph but typically around 10-15mph. Processing time is comparable to that of the dense stereo algorithm used [5] with minimal optimisation on a single core CPU. As this approach requires a high quality depth map the dense stereo algorithm used should be carefully considered as it can easily become the bottleneck for the processing at the resolutions used in this work. Results are presented in point cloud format showing before and after dynamic object removal. Figure 5.2 shows a typical road traffic scene with a vehicle preceding the camera, remaining in view for multiple stereo image pairs thus being reconstructed multiple times. Figure 5.8 demonstrates a different case (outlined in Section 5.3.2), where the moving object in the scene has constant depth, due to object motion being perpendicular to the camera motion, and cannot be reliably seen with disparity projections alone. The OFFD is used to align the intensity images and the moving objects identified in this channel. Figure 5.9 shows a slow moving object that has approximately 50% frame-to-frame overlap with itself, increasing the chance that intensity variation is not great

enough to be flagged as dynamic components, however it is still removed from the final point cloud. A large group of people in Figure 5.10 were removed primarily via the disparity projections approach as the motion had a large component towards the camera causing sufficient disparity variation. An interesting result, seen in Figure 5.11, a fast walking pedestrian moves approximately perpendicularly to the camera motion and the intensity projection method efficiently removes the dynamic components from the resulting point cloud. However, closer inspection of the scene, post removal, reveals the feet are still present in the point cloud. The feet of a walking pedestrian must be stationary when in contact with the ground thus they are temporarily part of the static background. The lower part of the leg that rotates about the ankle is not stationary therefore is successfully removed from the point cloud. The importance of the two difference maps varies with scenario. For moving objects that have a consistent apparent depth relative to the camera (i.e. moving perpendicularly to the camera motion) the intensity-difference maps are vital, as seen in Figure 5.4. In most on-road scenarios, with fast moving vehicles, the dominant motion is in a similar axis to the camera creating a large variation in frame-to-frame depth. With the disparity-difference stage of this technique using depth variation is becomes the primary mechanism for dynamic object removal.

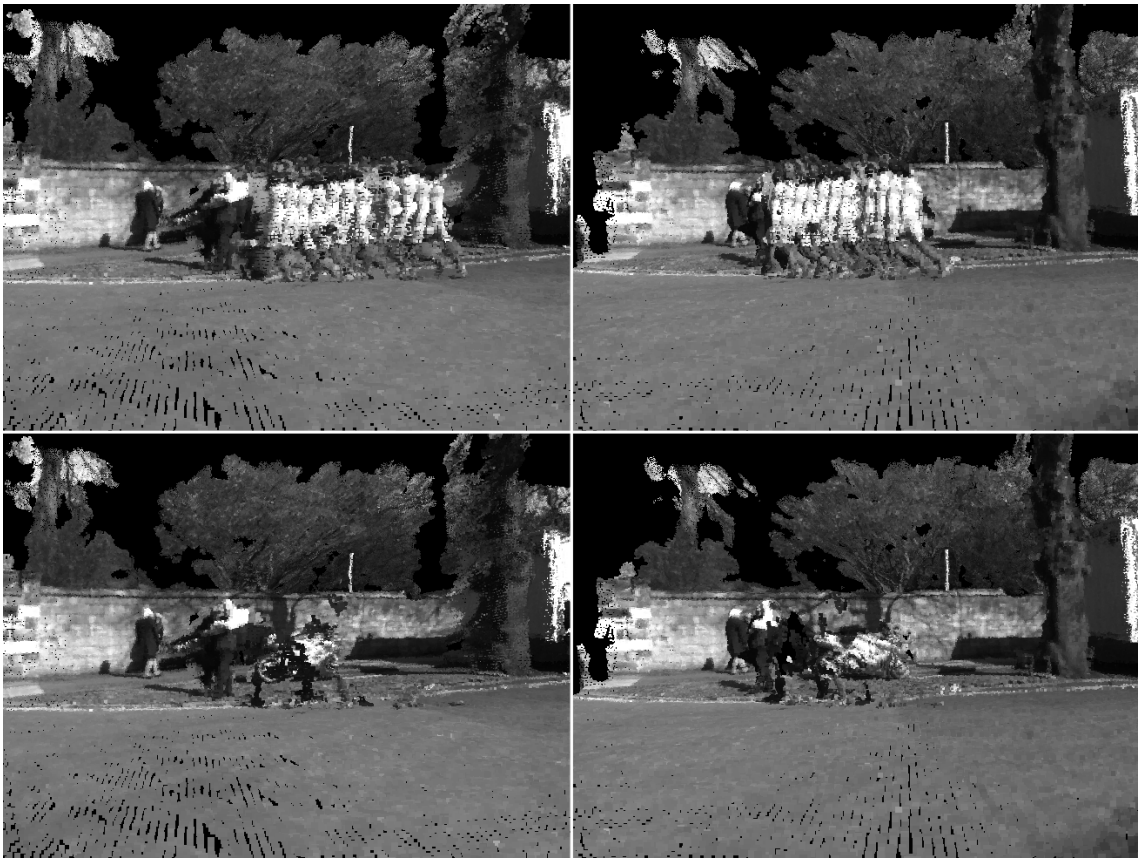

Figure 5.8: Left to Right: Two different viewpoints of the same point cloud. Top: Moving object reconstructed multiple times. Bottom: Moving object masked out of the reconstruction process.

Figure 5.9: Top: A slow moving pedestrian with approximately 50% self-overlap between frames. Bottom: Successfully removed from mapping solution.

Figure 5.10: Top: Large group of people moving at various speeds including some static bystanders. Bottom: Pedestrians are largely removed, elements that remained static between frames are still present.

Figure 5.11: Top: Fast moving pedestrian with motion perpendicular to camera motion. Bottom: Successful removal of moving components, however the feet remain visible in the point cloud as these are temporarily static during contact with the ground. The motion of the leg above the ankle has been successfully removed.

5.5 Conclusion

In this chapter we have demonstrated using stereo vision and an odometry solution we can re-use the disparity maps produced for the mapping solution as a significant processing stage in the removal of dynamic objects with minimal processing

overheads. High quality dense stereo and visual odometry provides synthetic dense optical flow information that is used to project intensity images captured at different times and poses into a common pose by constructing a virtual viewpoint. Projecting an intensity image into a virtual viewpoint equal to that of the prior image allows for accurate structure aware projective transforms facilitating the detection of temporal variances within the scene. Tuning the threshold of the disparity-difference map allows for a variable velocity discriminator and the intensity-difference map threshold allows for detection of image regions related to objects moving orthogonally. Unlike previous attempts that use sparse feature points [101, 103, 137] and computationally expensive segmentation algorithms [39–42, 104], our approach demonstrates accurate motion masks can be created in order to facilitate removal of dynamic objects from the final 3D scene map. This approach uses prior calculated dense disparity maps and the camera trajectory, already required for the 3D mapping process, which offers the ability to project frames to a common viewpoint for temporal pixel-wise analysis. This is illustrated upon on two different datasets, KITTI and our own, through varying camera motions and dynamic object characteristics. No assumptions have been made about the physical nature of the objects or the scene. An immediate extension to this work would be to quantify the effectiveness of the moving object removal using the annotation information supplied with the KITTI dataset. However, the annotation data of KITTI only supplies bounding boxes of objects, to better assess the performance of the moving object masks a dataset with manually labelled moving object masks would be required. Within our dataset, SGBM performed far beyond expectation [102] which led to the ability to compute highly accurate dense optical flow. The factors that made this dataset such a success, compared to the popular KITTI, will be the subject of future work.

5.6 Summary

We have demonstrated a novel process for generic dynamic object removal from 3D scene models created with a moving stereo camera. Through the use of existing data, required for the dense stereo 3D mapping pipeline, we remove dynamic scene compo-

nents regardless of object class. This extends prior work as we do not require extra computationally expensive processes like feature points [101, 103, 137] or foreground segmentation algorithms [39–42, 104]. By leveraging the camera pose information from SVO and the pixel-wise dense depth data we reproject depth and intensity images from different spatiotemporal locations into single spatial viewpoint to enable direct pixel-wise temporal comparison, exceeding the edge accuracy of [42, 101].

Chapter 6

Large Scale Reconstruction and Evaluation

In this section we outline the final processing stages, reconstruction results, data logging issues and repair, and evaluation. We show scale consistent global 3D model generation from SfM and dense stereo that rivals state of the art approaches in terms of reconstruction quality and exceeds them in 3D point generation rate.

6.1 Introduction

Final 3D reconstruction is performed using input from only multiple passive cameras, a mixture of stereo and monocular cameras with only the stereo cameras, by definition, having overlapping fields of view. We use our custom data collection platform, as detailed in Section 4.2.1, to obtain images from a moving vehicle. This chapter details some of the issues associated with capturing a custom dataset required for this type of reconstruction. We outline a method for synthesising intermediary images in order to repair (to a limited extent) some of the dataset by using work from Chapter 5 and applying it much the same way but utilising the results prior to the final output of moving-object-removal. Finally, we present images of the reconstruction output in the form of point clouds, demonstrating the high output resolution obtained from the SfM process and the multi-view reconstruction from non-overlapping cameras with automatic alignment.

6.2 Processing

At this stage we have the components to be able to perform high resolution SfM with very dense reconstruction from a side-facing camera (Chapter 4), dense 3D reconstruction and world-scaled odometry from a forward facing stereo camera (Chapter 3), and removal of dynamic objects from the scene observed with the forward facing stereo camera (Chapter 5). The final stage is combining the outputs from the SfM and stereo stages into a single reconstruction point cloud. With known relative positions of the cameras, but unknown rotations, we utilise the two pose-graphs (one from each processing pipeline) to perform an automatic alignment process that is made possible by the constraint that the image capture is synchronised between non-overlapping cameras and they are all rigidly connected to one another.

6.2.1 Automatic Alignment

An alignment stage is required to bring the individual point clouds from the non-overlapping cameras together into one large world-scaled point cloud correctly orientated to each other. This process of automatic alignment is performed online as the pose graphs of each camera are being constructed. At each additional new camera pose the process executes again with greater constraint than last time. We use a pose graph based approach as we can not guarantee that a given region of the scene will ever be viewed by multiple cameras in order to use feature based registration techniques. We therefore use the fact the cameras are rigidly mounted to the vehicle and their locations relative to each other is fixed. The work of [139] uses a vehicle fitted with four 182° fish-eye cameras mounted approximately orthogonal to one another around the vehicle. They perform VO from each camera during a sequence of several manoeuvres in an enclosed area with good features to track in all directions. While they do no make the assumption of overlapping views their approach does rely on instances of common views of the scene from different cameras. Physical properties of the vehicle, on which the cameras are mounted, can be used to constrain the expected motion. Work by [56] uses the knowledge of the mechanics of front-wheel steer vehicles and the properties of Ackermann steering [140]

as a prior to estimating possible motions of the camera. Both approaches successfully demonstrate calibration, however both have their limitations. Feature based alignment [139] assumes a shared field-of-view will exist at some point, a problem often encountered in our dataset where the narrow streets and parked cars present vastly different perspectives and scene obscuration by foreground objects. Complex vehicle manoeuvres require large areas in which the vehicle can freely move about and in an area with sufficient features to track. Systems that use properties of the vehicle's motion [56] as a basis for calibration rely on external information that can not be easily verified (e.g. wheel slippage creates a motion that does not conform to the expected constraints used for calibration). Using platform motion properties also restricts the use to similar designs and may not function on those that use drive-trains such as tracks or fixed axles.

The method presented here is independent of camera location on the platform and the platform type (in the limit of providing functional data for SfM and SVO). There is one constraint when it comes to motion. In order to properly calibrate the scale component of the SfM pose graph the initial motion of the platform must be straight forwards (or backwards). This arises from the fact we use SVO to obtain world-scaled motion of the platform (unsusceptible to wheel slip or driver-train configuration). Should the motion of the platform be rotating (vehicle yaw) at the start of the processing sequence, the cameras nearest to the point of rotation will experience a smaller radius of turn compared to cameras mounted a greater distance away. Therefore, the distance each camera has traversed is slightly different, hence scaling this motion magnitude by the SVO magnitude would be incorrect. An alternative method, to remove this initial motion constraint, is to have prior knowledge of the camera positions relative to the stereo camera being used for the SVO process. This allows us to take the pose estimation from SVO of the stereo camera and apply the appropriate offset for a given camera and construct an 'expected pose graph'. The local rotation of the camera does not have to be known as this is recovered in the alignment phase. This is the favoured approach, as the position of each camera on our camera mounting plate is well known because the mount was machined according to custom plans. Using this expected pose graph we

can fit the SfM estimated pose graph by minimising the following equation:

$$argmin \sum_{n=1}^{m} \left(\| [R|T] P_n^M - (P_n^S - C_{offset}) \|^2 \right) \tag{6.1}$$

Where P_n^M is n$^{\text{th}}$ image of the pose graph for the Monocular camera (SfM) or length m and P_n^S is the n$^{\text{th}}$ image of the Stereo pose graph generated from SVO of length m, C_{offset} is the known positional offset from the stereo camera to the monocular camera, and $[R|T]$ is the transform matrix required to align the two pose graphs. Using the Ceres solver [141] we solve the components of R and T to reduce the positional error between the expected position, calculated from SVO, and the estimated position from a camera from the SfM process. Ceres uses set of Schur-based solvers that use a Schur complement approach [141, 142].

Figure 6.1 illustrates the various pose graphs discussed previously. The expected path (green) is calculated with the an offset of the monocular camera from the forward facing stereo camera. The offset is known from the locations of the mounting holes in camera mounting plate. The monocular camera path estimated from the SfM process is calculated in its own local coordinate system, as a result, the monocular camera's motion is predominantly in X from a local origin *(0,0,0)*, shown as 'SfM Pose Graph' in Figure 6.1. Using the 'Expected Path' as our objective we compute the $[R|T]$ using Equation 6.1 that rotates 'SfM Pose Graph' to create 'SfM Aligned' (Figure 6.1). Results from this process can be seen in the multi-camera global reconstructions in Figures 6.11, 6.12, and 6.13. This alignment approach requires prior knowledge of the camera positions relative to the devices responsible for the platform odometry, in our case, the forward facing stereo cameras via SVO [63]. Camera rotations do not need to be known as they are estimated from this auto-alignment process. In order to eliminate rotational ambiguity along the axis of motion a turn is required to break the rotational symmetry of the pose graphs. Should the vehicle only drive forwards the alignment process would not be able to reliably estimate the monocular cameras roll angle relative to the vehicle. Our method assumes that images captured from all cameras are synchronised. If this approach is applied to unsynchronised image streams the image-to-image displacement will not be consistent

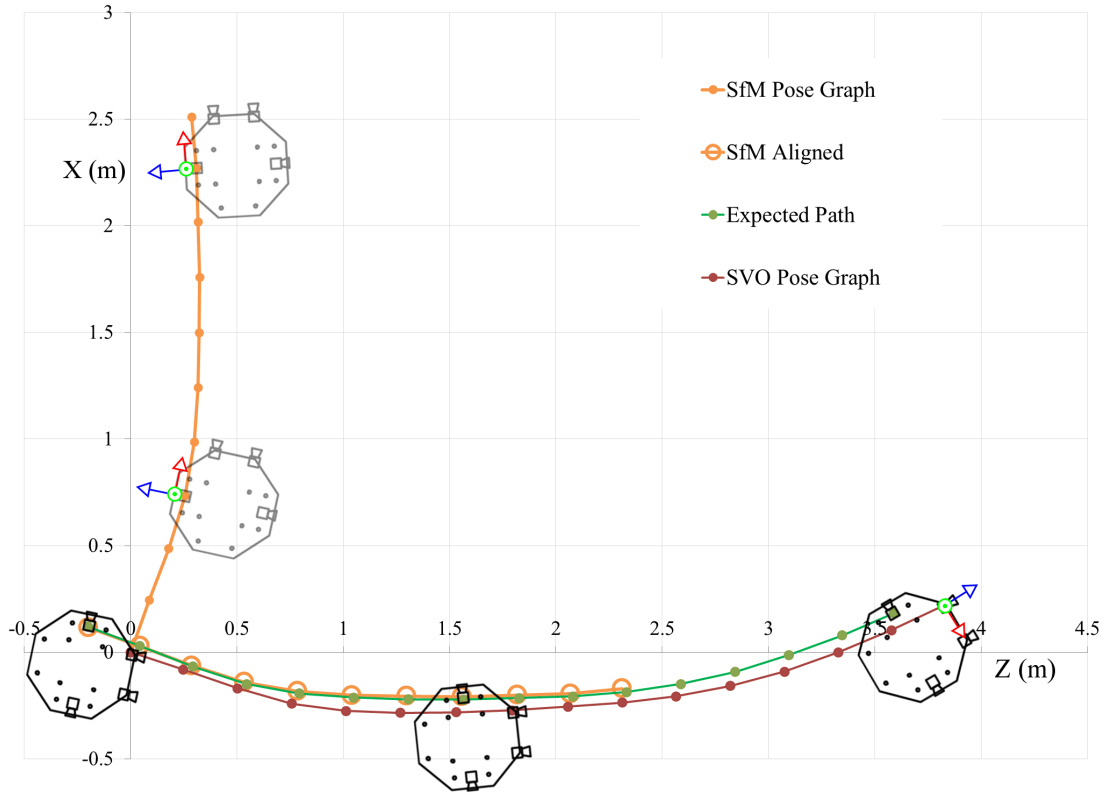

Figure 6.1: Pose graph alignment with camera mounting plate overlay for illustration presented in the stereo camera coordinate system.

across cameras giving the appearance of not being rigidly connected to each other. As we discuss in Section 6.2.2, synchronisation issues were encountered within our dataset, as a result this automatic alignment procedure was limited to only being able to be performed on short select sequences that remained synchronised.

6.2.2 Data Log Repair

Processing the data through our 3D reconstruction pipeline highlighted some unexpected errors with data collection. The processing technique detailed in the previous sections does not use any image features across the stereo images or side facing cameras to register or calculate relative camera poses. Instead we rely on the input images being synchronised across all cameras to allow for scale estimation through the SfM phase and online pose alignment between sets of cameras with non-overlapping views. During data collection it appears that there was an issue that caused the cameras to become unsynchronised. Time constraints prohibit extensive further

testing and isolation of the root cause of this synchronisation problem, elements for future investigations include the physical connection issues with vibration or an unseen software issue possibly caused by the cameras adjusting exposure settings at inappropriate times (i.e. when a new frame is requested). In the data logs this error is manifested as appearing as a dropped or missing frame in the image sequence, this is of primary concern for the forward facing stereo cameras and the SVO processing as a dropped stereo frame-pair can appear as a larger motion change that can in some cases cause the SVO process to fail.

Investigating the possible impact of dropped frames on SVO using our dataset is not possible, instead we test the SVO robustness on the KITTI data [34] by executing multiple runs of the same sequence with different probabilities that any given frame could be missing. We process a sequence of length 400 frames traversing a total distance of 322.6m with every frame included to obtain a ground truth. Using the same sequence we perform the same SVO process but include constant probabilities for each run of any given frame being dropped, $P_{drop} = \{0.05, 0.10, 0.15, 0.20, 0.25, 0.30\}$. Figure 6.2 shows the results from randomly dropped frames on a small sample from the KITTI dataset. We see that a mean end point positional error for $P_{drop} = \{0.05, 0.10, 0.15\}$ are all comparable at approximately 1.5m (0.46% of total odometry length). There is a significant increase for $P_{drop}(0.20)$, however most paths end near the ground truth path, with only one ending a significant distance away. Increasing the P_{drop} value, as expected, increases the size of the end point deviation and frequency. For nearly all cases shown in Figure 6.2 the SVO tracking error results in an increase in Z displacement, this can be attributed to the dropped frame occurring at the point of turn. Less rotational constraint information available during the turn results in under-rotation, causing the odometry path to run wide.

As outlined in Section 6.2.1, we require accurate positional data around a turn in order to calculate relative camera positions and enable accurate scaling and placement of our SfM solution. To make use of image sequences that have dropped frames, causing SVO failure and synchronisation issues, we use the process outlined in Chapter 5 to utilise the high quality dense depth information to create a synthetic frame to replace the missing data. When the stereo odometry processing of a new

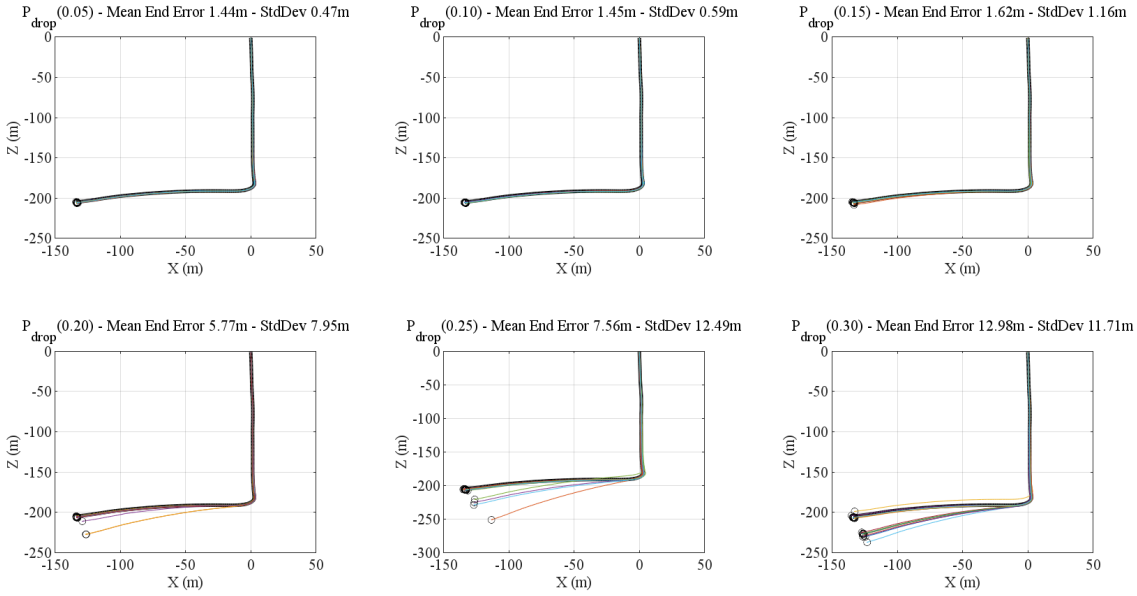

Figure 6.2: SVO performance in the presence of randomly dropped frames. Thick black line represents results from no dropped frames. Black circles indicate end positions of SVO runs. *Note: Axes are not equal scales.*

frame-pair fails to converge on a pose estimation the software initiates the repair process. Unlike in Chapter 5 where images were projected into previous coordinates, allowing differences to be detected and removed, we now project the last disparity image forwards, replicating the motion of the previous odometry step using the assumption that the vehicles frame-to-frame motion has not dramatically changed in that time.

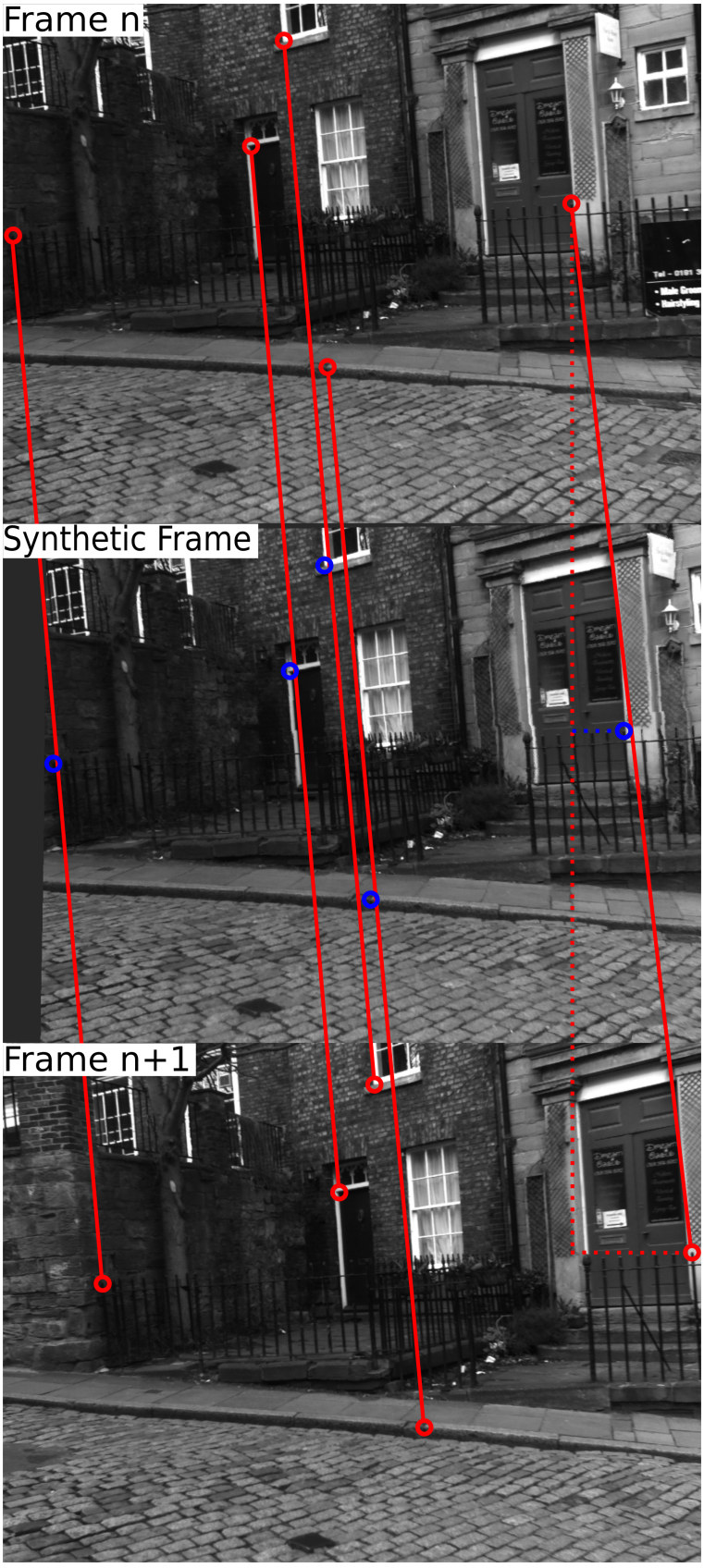

Figure 6.3: Top: Real frame (n) from left stereo camera at time T. Middle: Synthetic left stereo images at approximately $T + dt$. Bottom: Real frame $(n + 1)$ from left stereo camera at time $T + 2dt$.

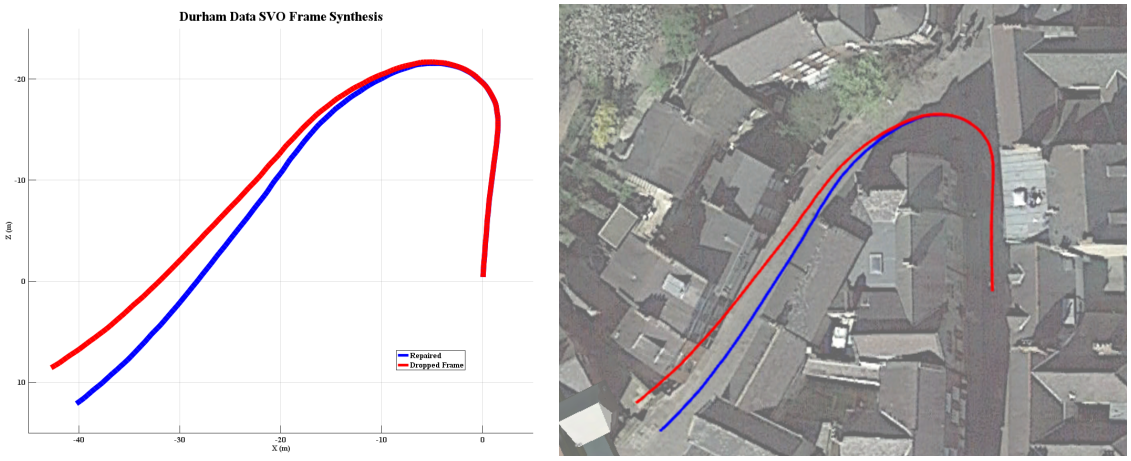

Figure 6.4: Left: Plot showing the different odometry paths with and without frame repair by image synthesis. Right: Overlay of camera paths on approximate location demonstrating the impact on positional error, map underlay from [1].

In, Figure 6.2.2, we highlight a few randomly selected interest points in each frame. The red circles show the position of the features in the real frames captured from the data logging platform. In this example the offset between the top image and bottom image is too great for the SVO algorithm to track enough features and is unable to estimate the camera pose. Using the dense stereo information and the odometry we can generate the missing frame (using the same approach at Chapter 5), thereby preserving the 3D nature of the scene in the synthetic image, allowing the SVO algorithm to successfully function under large pose changes.

In Figure 6.4 we show the effect of frame dropping and recovery using frame synthesis from dense stereo imaging. As illustrated previously, the path of the dropped frame SVO tends to run wide around a turn due to the lack of constraint on rotation. Without accurate ground truth positional information for our dataset, we are unable to perform quantitative analysis as to assess the level of improvement this repair process contributes. Full assessment may be possible using the KITTI data [34], however the approach from Chapter 5 requires pixel-wise density disparity maps. Obtaining such density using the KITTI data is not as trivial as from our own, comparing such examples created from the KITTI images using their own dense stereo algorithm (ELAS) [63] (Figure 5.3) to typical disparity maps generated using our Durham data with a well established dense stereo approach (SGBM) [5] (Figure 5.7). Further examples can be found in the Appendix.

6.3 Results

Throughout the thesis results pertinent to each Chapter have been presented therein. In this section we show the final reconstruction output from the whole processing chain. The following results are in the form of point cloud data rendered as such with no meshing or post processing to improve quality for viewing. Input images are sequences from our own data collection which was captured on various days around Durham, UK.

Figure 6.5: Top: Image from Google Earth [1]. Middle: Point cloud from SfM reconstruction viewed using Meshlab [6] (9.7 million points). Bottom: Same location captured an hour later with more stationary pedestrians present (11.2 million points).

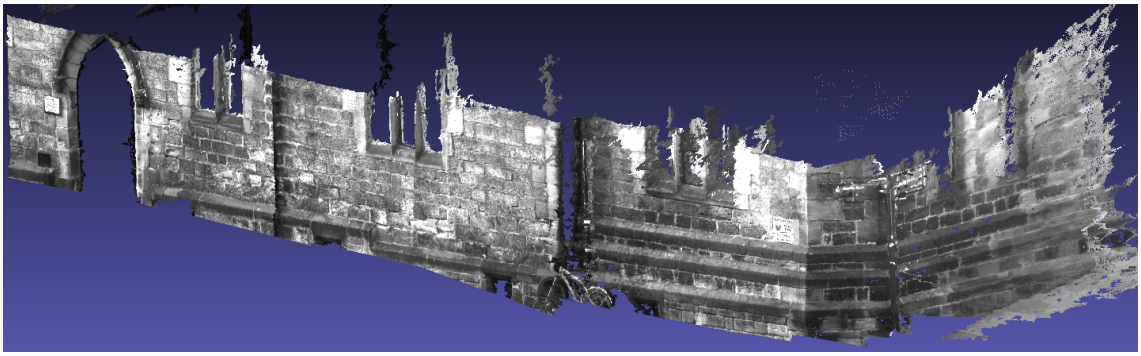

Figure 6.7: Orthographic projection of point cloud from North Bailey sequence (21.5 million points). *Note: this is a raw point cloud with no meshing or texturing.*

Figure 6.6: A close up render of the same point cloud in Figure 6.5 of the region highlighted with the red square, demonstrating the fine detail of the reconstruction.

Figure 6.11 shows the real-world scaled reconstructions from non-overlapping cameras using two different approaches. Each image provides different viewpoints of the same point cloud. The left side of the point cloud being reconstructed with SfM (Chapter 4) and the right side using SGBM [5] with SVO [63]. Registration and alignment is performed using the approach outlined in Section 6.2.1. The lower left image highlights (red circle) the curbside and double-yellow parking restriction lines in both SfM and stereo reconstructions, showing good positional and rotational agreement of the point clouds and therefore the automatic-alignment process.

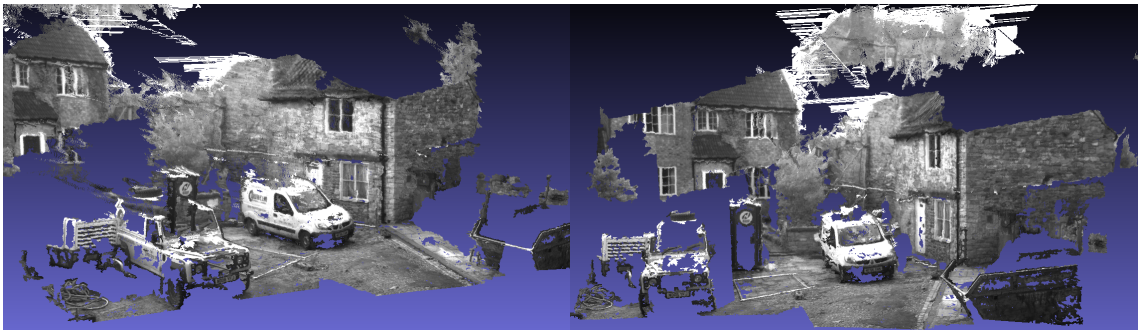

Figure 6.8: Reconstruction from South Bailey, Durham, UK.

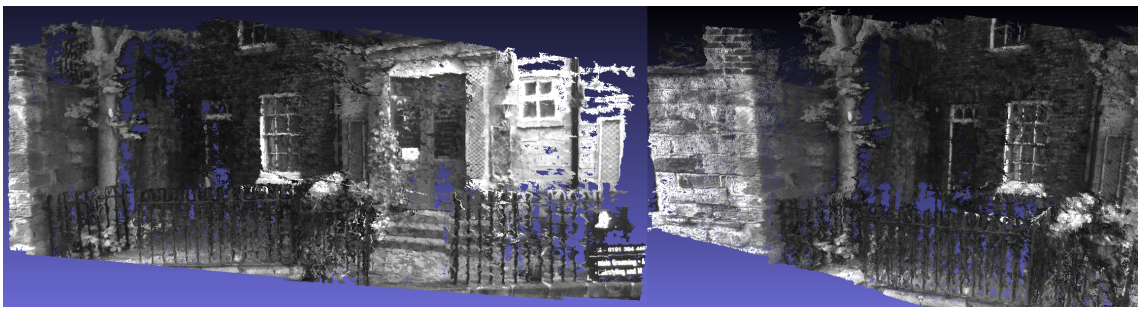

Figure 6.9: SfM densification showing fine repeating structure of railings being accurately reconstructed.

Figure 6.12 shows the surround view reconstruction created from three non-overlapping cameras (monocular left, right and forward facing stereo). The automatic alignment procedure appears to have been effective here at registering the relative positions of each camera.

6.4 Evaluation

Direct comparison of reconstruction quality and speed on different datasets and hardware using a variety of techniques is difficult without a dense ground truth. Using the reconstruction rate metric of seconds per camera and average number of 3D reconstructed points per second we show that our approach is two orders of magnitude faster than state of the art in terms of generating unique 3D point measurements. Our approach runs on a single thread of a 2012 Intel i7-3610QM (3.3GHz max) CPU on a consumer laptop. The number of features used for the bundle-adjustment and the processing strategy, whether all points are processed or a subset, impacts the run-time and the memory footprint. Using a nearest-neighbor

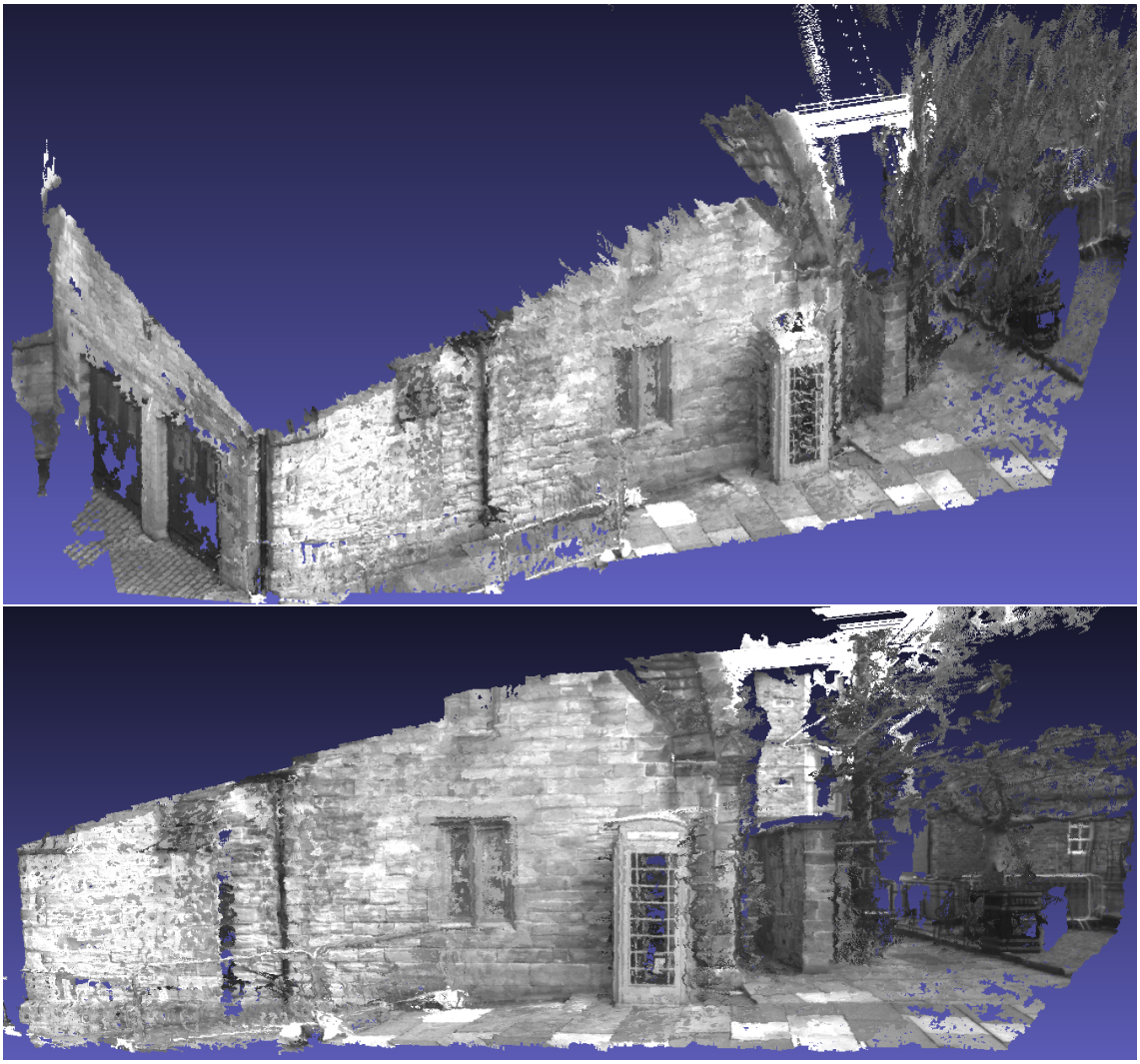

Figure 6.10: Top: Orthographic projection of dense SfM reconstruction. Bottom: Perspective projection of same point cloud from a different virtual camera position.

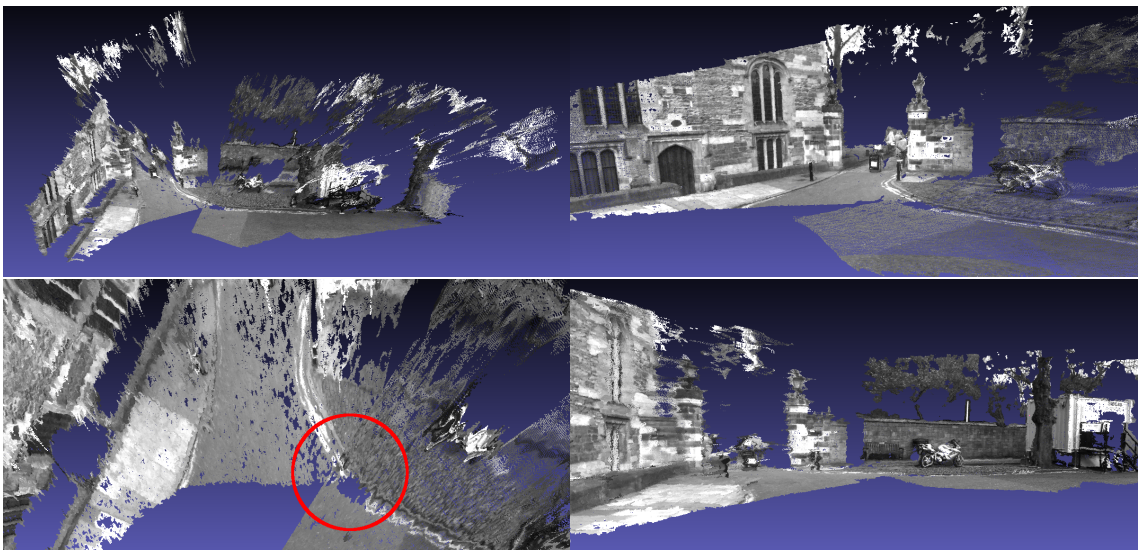

Figure 6.11: Combination point cloud created from dense SfM (Chapter 4) and dense stereo [5].

Figure 6.12: Top: Google Street View [7] image showing the North Bailey location. Bottom: Reconstruction from 3 non-overlapping views auto-aligned.

Figure 6.13: Left: Ground level view of aligned reconstruction. Right: Birdseye view of the same combined point cloud.

radius of 20px on our Durham Cathedral sequence and a full point BA strategy the memory load on the system fluctuates however it never exceeded 680MB, a typical maximum we see throughout our evaluation on similar length sequences.

Results in Table 6.1 shows reconstruction information for two different approaches that demonstrate dense outdoor reconstruction. In this case we are considering the reconstruction rate with two metrics, processing time per camera registered in the SfM process and the number of 3D point reconstructions per second. Taken in isolation, our sparse SfM approach is slower than [38] and only reconstructs approximately the same number of 3D points per second as the unordered approach of [37]. However, with the addition of the dense processing stage we get bursts of highly dense 3D data that provides millions of new 3D measurements in a very short time. Every execution of the densification process can potentially generate 1.2 million 3D measurements from the 1280×960px images, depending on the performance of the dense stereo approach used. As a consequence, our approach has been seen to effectively generate an average in excess of 32k 3D measurements per second, producing the almost pixel wise dense point clouds presented in this work. Commercial software package, Photoscan [2], is the next best in terms of the raw number of 3D data points generated per unit processing time at approximately an order of magnitude less than ours. It does however outperform our approach in accuracy terms. Figure 6.14 shows the same sequence of 50 images reconstructed using our SfM approach and Photoscan, both show similar levels of reconstruction have been achieved. Processing in a global manner affords the software the opportunity to revisit parts of the reconstruction from different cameras with variable

Result	Work	Dataset	Cameras	Points	Time	t/c	n/t	Fig.
	[37]	Dubrovnik	4,585	498,982	81,000	17.67	6.16	
	[37]	Rome	2,097	2,712,301	75,600	36.05	35.88	
	[37]	Venice	13,699	6,119,207	234,000	17.08	26.15	
	[38]	Loop	4,342	1,101,515	3,251	0.75	338.82	
	[38]	St. Peter's	1,267	292,379	583	0.46	501.51	
	[38]	Colosseum	1,157	293,724	591	0.51	496.99	
1	Ours (S)	North Bailey	100	5,336	305	3.05	17.50	
2	Ours (D)	North Bailey	100	21,460,442	654	6.54	32814.13	6.7
3	Ours (D)	North Bailey	50	10,514,541	762	15.24	13798.61	4.26
	[2]	Market Place	50	6,157,214	2309	46.18	2666.76	6.14
4	Ours (S)	Market Place	50	8,708	342	6.84	25.46	
5	Ours (D)	Market Place	50	8,884,743	486	9.72	18281.36	6.14
6	Ours (D)	Barbour Shop	70	17,218,089	1822	26.03	9450.10	6.9
7	Ours (D)	Garage	60	12,948,602	486	8.10	26643.21	6.10
8	Ours (D)	South Bailey	30	4,323,404	186	6.20	23244.11	6.8
9	Ours (D)	Library	50	11,179,380	253	5.06	44187.27	6.5

Table 6.1: Results showing comparison between different methods in terms of reconstruction rate. The S and D in parenthesis denotes which version of our approach is being used, S referring to the sparse feature point based reconstruction, D is with the addition of the dense stereo-from-motion phase. Time is processing time in seconds, t/c is in seconds per camera and n/t is reconstructed 3D points per second. *Note: results set 2 used a nearest-neighbor radius of 40px and results set 9 used nearest-neighbor radius of 20px but used a faster Intel i7-6700K CPU.*

baselines between reconstructing pairs. As demonstrated in Figure 4.29, the baseline of the cameras used for reconstruction has a significant impact on the reconstruction quality.

We achieve this by leveraging the strong spatiotemporal ordering of our input data. Using a lightweight SfM approach that exploits the fact our input images are sequential and therefore are constrained to limited motion changes, we construct a pose graph and 3D scene using a sparse but uniformly distributed point set. Using bundle adjustment to optimise camera pose graphs over sparse point sets improves the camera-to-camera pose information, allowing us to use standard stereo rectification methods to obtain a stereo pair from a single moving camera. Processing this, now rectified stereo-from-moving-monocular, image pair with conventional dense stereo approaches produces a dense disparity image resulting in a vast number of 3D point triangulation points in very little time. While our approach may tackle

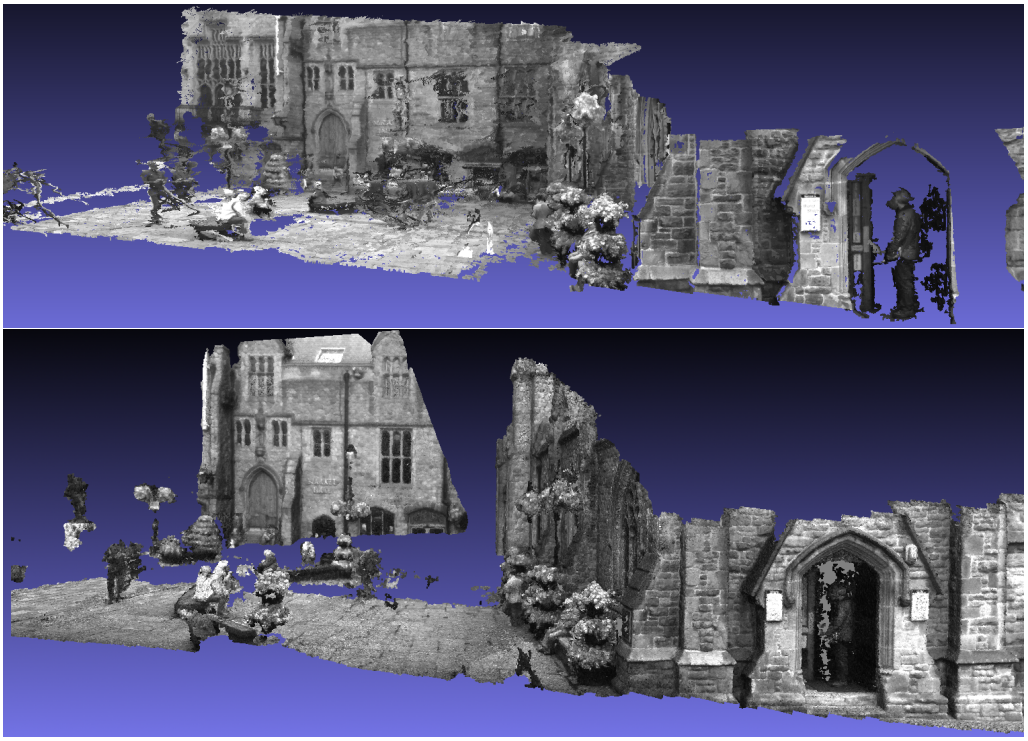

Figure 6.14: Top, partial reconstruction of Durham Marketplace using our SfM approach. Bottom, same sequence reconstructed using a leading commercial package [2]

the issue of reconstruction in a less general sense, requiring sequential images from an approximately side facing camera, it does address the need to be able to densely reconstruct a scene with as few images as possible, as a second observation of the scene may not be possible.

6.5 Summary

In this chapter we have combined the results of dense stereo evaluation and SVO from Chapter 3 and the SfM process outlined in Chapter 4 to create correctly orientated and scale consistent point clouds (in the limit of data synchronisation errors) from non-overlapping cameras rigidly mounted on a moving platform. We have shown that the qualitative reconstruction performance of our SfM approach rivals that of leading commercial photogrammetry software [2] but achieves this in a significantly shorter time. Our approach also exceeds the 3D point generation rate of prior work [37,38] by at least an order of magnitude (Table 6.1).

Chapter 7

Conclusions and Discussion

In this chapter we conclude our findings and discuss the contributions made to state of the art. We outline the limitations of the work, the implication and impact to industry, and possible topics for future research.

7.1 Conclusions

It is clear that over the years many have attempted to solve the problem of 3D mapping from a vehicle with a number still pursuing this objective as it remains an open problem. The work carried out in this project has demonstrated the ability to create high resolution 3D models around a moving vehicle using relatively low cost hardware simply mounted on a vehicle with minimal placement requirements.

Using a novel stereo assessment method we concluded that despite the vast amount of attention dense stereo algorithms have received [34], their advancements over simpler and faster approaches are not necessarily applicable to real-world use in challenging automotive environments [36, 102]. In addition, the current benchmarking suites are not an unbiased source of data for testing dense stereo approaches, as our disparity map results on our own dataset show the baseline and focal length is a significant factor in how effective a given algorithm is at creating a dense depth map. Therefore, in performance ranking of dense stereo algorithms the end application should play a large part in the decision process.

As previously discussed, using stereo vision for 3D mapping has the advantage

of using images sampled at the same time therefore having the ability to reconstruct dynamic objects, such as cars and pedestrians, a vital task for object avoidance. However, the consequence of this is any given dynamic objects are also observed at multiple spatial locations resulting in ghosting or multiple instances being created in the final map. We demonstrated that, unlike other solutions [39–42], minimal extra image processing is required to extract the information to perform removal of dynamic objects from the 3D scene. By utilising the data already acquired for the main task of 3D reconstruction, camera pose and depth, we are able to remove dynamic objects with fewer processing steps than previous work [104].

The configuration of the cameras on our test platform allowed for the creation of a unique Structure-from-Motion process that could perform online densification of the sparse point cloud from the SfM process. While there already exists many SfM pipelines that can perform real-time mapping [28, 29] they often achieve this performance at the cost of resolution using downsampled images, typically in the region of 640×480px. We use considerably higher image sizes of 1280×960px, a four-fold increase in the number of pixels. By using an optical flow based approach to feature tracking we avoid the problem of matching descriptors across many historic frames and by design are able to easily index all previous track points associated with a given point in the current frame [38], thus allowing for a dynamic bundle-adjustment window size. A qualitative comparison with a leading commercial photogrammetry software package, Photoscan [2], illustrates that reconstruction is comparable in terms of density, while run-time of our pipeline is 486s and [2] achieves similar results in 2309s using all 8-threads of our CPU whereas our approach is not optimised and only utilises a single thread.

7.2 Contributions

In this work we have demonstrated several key contributions towards developing a viable alternative to expensive laser scanners for on vehicle 3D mapping using low cost, small cameras with minimal power requirements.

- We presented a new approach to assessing the performance of dense stereo

algorithms on automotive data in an object wise manner (Chapter 3). Leading to the conclusion that well established approaches, namely that of [5], still achieve comparable levels of accuracy to that seen in more modern approaches like [3] on real-world automotive data.

- In Chapter 4 we demonstrated a new approach to dense SfM that facilitates generation of large amounts of 3D data that exceeds the point generation rate of other approaches [37, 38] by 2 orders of magnitude (Table 6.1) on real outdoor data.

- An efficient reuse of data led to a novel approach to tackle the problem of dynamic object removal by reusing existing data from the mapping process that requires minimal extra image processing (Chapter 5).

- Finally, we show how using SVO coupled with SfM captured from non-overlapping cameras and using pose graph alignment can result in scale consistent surround view mapping (Chapter 6).

7.2.1 Industrial Impact

In this work, we have demonstrated that high quality 3D reconstructions can be achieved with forward facing stereo vision coupled with a single side facing cameras with no common field of view. The implications for the automotive industry are that small discreet cameras can be installed in the B-pillar of a vehicle and still be used to generate 3D maps of the area around a moving vehicle. Low cost automotive grade cameras are becoming ubiquitous on modern vehicles whereas other depth sensing devices such as LIDAR are the reserve of research vehicles due to high unit costs and, as previously discussed, are limited to locations they can be situated. While this work serves as a successful proof of concept it is far from being deployable in commercial vehicles, requiring further optimisation and testing in a greater variety of scenarios, environments and, importantly, weather conditions.

Visual odometry is a mature solution that, coupled with existing integrated GPS receivers, could provide a positioning system robust to GPS dropouts likely to happen in built up areas. Our moving object removal approach requires minimal

extra processing, only needing to process existing pose and 3D information, avoiding other computationally expensive image processing techniques, as previously discussed, such as segmentation or optical flow. In this work we used our approach to remove dynamic scene components from the final reconstructed 3D model. Inverting the output, we could apply this technique in the same manner to identify regions of the image that contain dynamic components. As the output of our approach is not dependent on object class i.e. car, pedestrian, pram, horse, etc. it therefore provides flexibility that is necessary for real world applications.

7.2.2 Limitations

While the 3D point generation rate does exceed that of prior work, our approach is still not real-time processing on a consumer grade laptop. It is however considered an online process. If a platform was traversing a given scene, at slow enough speeds, the software could incrementally build the 3D point clouds without knowledge of the total sequence length. The current implementation only utilises a single core of the CPU, this greatly limits the potential speed of processing using modern CPU's that regularly contain four cores. This version does not exploit the GPU for added acceleration.

The final source of limitation with this project lies in the capabilities of the data capture platform. There are several publicly available datasets that focus on automotive environments, however none offered the surround view coverage or image resolution required to perform surround-view 3D mapping. As a result, a custom platform had to be created from scratch. Creating an initial data-capture platform that was lightweight and able to record synchronised stereo frames (640×480px each) and 3D data from a Microsoft Kinect, a structured light depth camera, proved to be a fairly straightforward task. It was fitted to a small robotic platform to facilitate single user data capture with minimal safety considerations or road legislation to worry about. While this was suitable as a proof of concept and an early developmental platform, it requires a substantial upgrade to be of use for large scale data capture on the automotive scale. Both upgrading the existing cameras and adding two more, side facing, imagers dramatically increased the bandwidth

requirements. The existing USB host controller was able to cope, however, the spinning disk hard-drive did not have the write speed to save the incoming data at the frame acquisition rate, resulting in an upgrade to a solid-state drive (SSD). The addition of a second SSD in a RAID 0 configuration ensured plenty of write speed to cope with more cameras. Estimations of hardware capabilities and expected data rates only provide an indication of bandwidth capacity, with chipset, driver, and OS compatibility overheads proved difficult to anticipate. The final step of data capture was to synchronise the start of image acquisition on each camera, this was a vital part of our processing pipeline that allowed us to scale the SfM results to the output from the SVO stage therefore creating world scaled SfM. Synchronisation was controlled by an external clock signal from an Arduino microcontroller triggering the GPIO (general purpose input output) pin on each cameras simultaneously with a +5v pulse.

The end result was a platform able to capture image data from six cameras (stereo forwards, stereo rear, monocular left, monocular right) at 1.3 megapixels each synchronised at 5Hz or four cameras (stereo front, monocular left, monocular right) at 7.5Hz. The four camera configuration at 7.5Hz was most commonly used. A large portion of the time the data collection was successful, however there were instances where synchronisation failed for an unknown reason and remained undetected until very late in the project. Timing errors often resolved themselves bringing the video sequence back into sync. The primary suspect of the timing errors is the Arduino trigger and a loose connection caused by vehicle vibrations. Other sources could have been a race-condition in the multithreading logging code, a bottleneck or write buffer somewhere in the OS, or auto exposure instructions from the camera drivers commanding a camera to adjust settings such as exposure time (shutter time) causing an interrupt in the triggering.

7.3 Further Work

We have demonstrated the ability to create high quality 3D reconstructions from multiple low-cost non-overlapping cameras mounted on a moving vehicle. Despite

the success there remain some areas where this work can be expanded on.

7.3.1 Data Capture

As detailed in Section 7.2.2 the errors from data collection severely restricted the length of sequences that could be successfully processed. Work is currently in progress to design a new data collection platform building on knowledge gained and the discovered requirements of this project.

7.3.2 Dense Stereo

Dense stereo requires some further investigation into failure modes and success cases. As mentioned previously the performance of SGBM [5] far exceeded expectations on our dataset compared to its performance on KITTI. A study where the baseline and field of view are varied in a controlled way would provide valuable information as to how various algorithms degrade as a function of these physical properties. Furthermore, evaluation over the algorithms parameter space (e.g. block sizes, window sizes, cost functions, uniqueness ratios, etc.) would reveal the limitations and extent to which a given algorithm functions as expected. A full parameter evaluation would be a large body of work, however it could be automated easily as an offline process. Variations of the physical parameter space (baseline) would require some careful and considered mechanical engineering to ensure optical calibration would not be required at every tested baseline. Every execution of the calibration process introduces possible differences in lens distortion characterisation, which could result in masking the performance results of the dense stereo algorithm with results of the calibration process.

7.3.3 Structure from Motion Pipeline

While we have shown that this approach to SfM can produce pixel-wise dense points clouds it would benefit from optimisation to improve the real-time performance. As the approach was built around using patch based optical flow techniques it provides great opportunity for acceleration via a GPU. Further ways to extend this work

would be to implement the latest meshing and texturing techniques [96, 143] to create better looking models for either offline viewing or calculating how objects may interact with the environment (e.g. traversability paths, gradients of paths or regions that may be susceptible to pooling of liquids and therefore are likely to have standing water).

7.3.4 Extend Coverage

The final component that would extend work would be to capture and create the full 360° of coverage around a vehicle in 3D. Due to data collection difficulties, reconstruction was limited to approximately 270° (front, left, right).

Bibliography

[1] Google, "Google Earth," 2016. [Online]. Available: https://www.google.com/earth/

[2] Agisoft, "PhotoScan," 2017. [Online]. Available: http://www.agisoft.com/

[3] A. Geiger, M. Roser, and R. Urtasun, "Efficient Large-Scale Stereo Matching," *Asian Conference on Computer Vision*, pp. 25–38, 2011.

[4] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," *Lecture notes in computer science*, vol. 3951, p. 14, 2006.

[5] H. Hirschmüller, "Stereo processing by semiglobal matching and mutual information." *IEEE transactions on pattern analysis and machine intelligence*, vol. 30, no. 2, pp. 328–41, 2008.

[6] P. Cignoni, P. Cignoni, M. Callieri, M. Callieri, M. Corsini, M. Corsini, M. Dellepiane, M. Dellepiane, F. Ganovelli, F. Ganovelli, G. Ranzuglia, and G. Ranzuglia, "MeshLab: an Open-Source Mesh Processing Tool," *Sixth Eurographics Italian Chapter Conference*, pp. 129–136, 2008.

[7] Google, "Google Maps," 2017. [Online]. Available: https://www.google.co.uk/maps

[8] A. Giusti, J. Guzzi, D. C. Cire, F.-l. He, J. P. Rodríguez, F. Fontana, M. Fässler, C. Forster, J. Schmidhuber, G. D. Caro, D. Scaramuzza, and L. M. Gambardella, "A Machine Learning Approach to Visual Perception of Forest Trails for Mobile Robots," *IEEE Robotics and Automation Letters*, vol. PP, no. 99, pp. 1–1, 2015.

[9] W. Van Der Mark and D. M. Gavrila, "Real-Time Dense Stereo for Intelligent Vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 1, pp. 38–50, 2006.

[10] A. Howard, L. H. Matthies, A. Huertas, M. Bajracharya, and A. Rankin, "Detecting pedestrians with stereo vision: safe operation of autonomous ground vehicles in dynamic environments," *Proceedings of the 13th International Symposium of Robotics Research*, 2007.

[11] I. Katramados and T. P. Breckon, "Real-time visual saliency by Division of Gaussians," in *International Conference on Image Processing*, 2011, pp. 1701–1704.

[12] N. Haala, M. Peter, J. Kremer, and G. Hunter, "Mobile Lidar Mapping for 3D Point Cloud Collection in Urban Areas – a Performance Test," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, pp. 1119–1124, 2008.

[13] A. Teichman and S. Thrun, "Practical object recognition in autonomous driving and beyond," in *Advanced Robotics and its Social Impacts*. IEEE, 2011, pp. 35–38.

[14] F. Zhang, H. Stahle, G. Chen, C. Simon, C. Chen, C. Buckl, and A. Knoll, "A sensor fusion approach for localization with cumulative error elimination," *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, pp. 1–6, 2012.

[15] A. Geiger, F. Moosmann, O. Car, and B. Schuster, "Automatic camera and range sensor calibration using a single shot," in *International Conference on Robotics and Automation*. IEEE, 2012, pp. 3936–3943.

[16] J. Levinson and S. Thrun, "Automatic Online Calibration of Cameras and Lasers," in *Proceedings of Robotics: Science and Systems*. Robotics: Science and Systems Foundation, 2013, pp. 24–28.

[17] S. Ullman, *The Interpretation of Visual Motion*. MIT Press, 1979.

[18] C. Tomasi and T. Kanade, "Shape and motion from image streams: a factorization method." *Proceedings of the National Academy of Sciences*, vol. 90, no. 21, pp. 9795–9802, 1992.

[19] R. I. Hartley, "Theory and practice of projective rectification," *International Journal of Computer Vision*, vol. 35, no. 2, pp. 115–127, 1999.

[20] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle Adjustment - A Modern Synthesis," *International Workshop on Vision Algorithms*, vol. 34099, pp. 298–372, 1999.

[21] M. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, and M. Csorba, "A solution to the simultaneous localization and map building (SLAM)," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 3, pp. 229–241, 2001.

[22] D. Scharstein and R. Szeliski, "A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms," *International journal of computer vision*, vol. 47, no. 1-3, pp. 7–42, 2001.

[23] A. Z. Richard Hartley, *Multiple View Geometry*, second edi ed. Cambridge University Press, 2003, vol. 53, no. 9.

[24] A. J. Davison, "Real-time simultaneous localisation and mapping with a single camera," in *International Conference on Computer Vision*, vol. 2. IEEE, 2003, pp. 1403–1410.

[25] D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry," *Computer Vision and Pattern Recognition*, vol. 1, pp. 652–659, 2005.

[26] N. Snavely, S. M. Seitz, and R. Szeliski, "Photo tourism," *ACM Transactions on Graphics*, vol. 25, no. 3, p. 835, 2006.

[27] Pix4D SA, "Pix4d," 2017. [Online]. Available: https://pix4d.com/

[28] R. a. Newcombe, S. J. Lovegrove, and A. J. Davison, "DTAM: Dense tracking and mapping in real-time," *International Conference on Computer Vision*, pp. 2320–2327, 2011.

[29] J. Engel, T. Sch, and D. Cremers, "LSD-SLAM: Large-Scale Direct Monocular SLAM," *European Conference on Computer Vision*, pp. 1–16, 2014.

[30] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski, "Towards internet-scale multi-view stereo," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1434–1441, 2010.

[31] J. L. Schönberger and J.-M. Frahm, "Structure-from-Motion Revisited," *Computer Vision and Pattern Recognition*, pp. 4104–4113, 2016.

[32] A. Akbarzadeh, J.-M. Frahm, P. Mordohai, B. Clipp, C. Engels, D. Gallup, P. Merrell, M. Phelps, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewenius, R. Yang, G. Welch, H. Towles, D. Nister, and M. Pollefeys, "Towards Urban 3D Reconstruction from Video," *Third International Symposium on 3D Data Processing, Visualization, and Transmission*, pp. 1–8, 2006.

[33] H. Hirschmuller and D. Scharstein, "Evaluation of Cost Functions for Stereo Matching," in *Conference on Computer Vision and Pattern Recognition*. IEEE, 2007, pp. 1–8.

[34] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3354–3361.

[35] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nešić, X. Wang, and P. Westling, "High-Resolution Stereo Datasets with Subpixel-Accurate Ground Truth," in *Lecture Notes in Computer Science*, 2014, vol. 8753, no. 1, pp. 31–42.

[36] F. Mroz and T. P. Breckon, "An empirical comparison of real-time dense stereo approaches for use in the automotive environment," *European Association for Signal Processing Journal on Image and Video Processing*, vol. 2012, no. 1, p. 13, 2012.

[37] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski, "Building Rome in a day," in *International Conference on Computer Vision*. IEEE, 2009, pp. 72–79.

[38] C. Wu, "Towards linear-time incremental structure from motion," in *International Conference on 3D Vision*, 2013, pp. 127–134.

[39] A. Kundu, K. M. Krishna, and J. Sivaswamy, "Moving object detection by multi-view geometric techniques from a single camera mounted robot," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2009, pp. 4306–4312.

[40] A. Kundu, K. M. Krishna, and C. V. Jawahar, "Realtime motion segmentation based multibody visual SLAM," *Proceedings of the Seventh Indian Conference on Computer Vision, Graphics and Image Processing*, pp. 251–258, 2010.

[41] J. Klappstein, T. Vaudrey, C. Rabe, A. Wedel, and R. Klette, "Moving object segmentation using optical flow and depth information," in *Pacific Rim Symposium on Image and Video Technology*, 2009, pp. 611–623.

[42] P. Lenz, J. Ziegler, A. Geiger, and M. Roser, "Sparse scene flow segmentation for moving object detection in urban environments," in *IEEE Intelligent Vehicles Symposium*. IEEE, 2011, pp. 926–932.

[43] L. Fabian, S. Kalyan, H. Sunil, and G. Michael, "Shading-Aware Multi-view Stereo," in *European Conference on Computer Vision*, vol. 9905, no. 1. Springer International Publishing, 2016, pp. 35–35.

[44] J. Han, L. Shao, D. Xu, and J. Shotton, "Enhanced computer vision with Microsoft Kinect sensor: A review," *IEEE Transactions on Cybernetics*, vol. 43, no. 5, pp. 1318–1334, 2013.

[45] A. Fossati, J. Gall, H. Grabner, X. Ren, and K. Konolige, *Consumer Depth Cameras for Computer Vision*, 1st ed., A. Fossati, J. Gall, H. Grabner, X. Ren, and K. Konolige, Eds. Springer London, 2013.

[46] A. Kolb, E. Barth, R. Koch, and R. Larsen, "Time-of-flight cameras in computer graphics," *Computer Graphics Forum*, vol. 29, no. 1, pp. 141–159, 2010.

[47] P. Fankhauser, M. Bloesch, D. Rodriguez, R. Kaestner, M. Hutter, and R. Siegwart, "Kinect v2 for mobile robot navigation: Evaluation and modeling," in *International Conference on Advanced Robotics*. IEEE, 2015, pp. 388–394.

[48] W. H. Long, D. H. Mooney, and W. A. Skillman, "Pulse doppler radar," *Radar Handbook*, pp. 17.1–17.42, 1990.

[49] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski, "A comparison and evaluation of multi-view stereo reconstruction algorithms," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 519–528, 2006.

[50] K. Konolige, "Small Vision Systems: Hardware and Implementation," *Robotics Research*, pp. 203–212, 1998.

[51] C. Unger, C. Unger, S. Benhimane, S. Benhimane, E. Wahl, E. Wahl, N. Navab, and N. Navab, "Efficient Disparity Computation without Maximum Disparity for Real-Time Stereo Vision," in *British Machine Vision Conference*, 2003, pp. 1–12.

[52] J. Lu, K. Zhang, G. Lafruit, and F. Catthoor, "Real-time stereo matching: A cross-based local approach," *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 733–736, 2009.

[53] L. Wang, M. Liao, M. Gong, R. Yang, and D. Nister, "High-quality real-time stereo using adaptive cost aggregation and dynamic programming," *Third International Symposium on 3D Data Processing, Visualization, and Transmission*, vol. 67, pp. 798–805, 2007.

[54] Q. Yang, "A non-local cost aggregation method for stereo matching," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, no. 1, pp. 1402–1409, 2012.

[55] D. Scaramuzza, F. Fraundorfer, M. Pollefeys, and R. Siegwart, "Absolute scale in structure from motion from a single vehicle mounted camera by exploiting nonholonomic constraints," *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1413–1419, 2009.

[56] D. Scaramuzza, F. Fraundorfer, and R. Siegwart, "Real-time monocular visual odometry for on-road vehicles with 1-point RANSAC," *IEEE International Conference on Robotics and Automation*, pp. 4293–4299, 2009.

[57] D. M. Helmick, Y. Cheng, D. S. Clouse, M. Bajracharya, L. H. Matthies, and S. I. Roumeliotis, "Slip compensation for a mars rover," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1419–1426, 2005.

[58] M. G. Wing, A. . Eklund, and L. D. Kellogg, "Consumer-Grade Global Positioning System (GPS) Accuracy and Reliability," *Journal of Forestry*, vol. 103, no. 4, pp. 169–173, 2005.

[59] S. Song and M. Chandraker, "Robust Scale Estimation in Real-Time Monocular SFM for Autonomous Driving," in *IEEE International Conference on Intelligent Robots and Systems*, 2014, pp. 1566–1573.

[60] C. Kee and B. Parkinson, "Wide area differential GPS (WADGPS): future navigation system," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 32, no. 2, pp. 795–808, 1996.

[61] C. Harris and M. Stephens, "A Combined Corner and Edge Detector," *Procedings of the Alvey Vision Conference 1988*, pp. 147–151, 1988.

[62] Z. Zhang, "A Flexible New Technique for Camera Calibration," *Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[63] A. Geiger, J. Ziegler, and C. Stiller, "StereoScan: Dense 3d reconstruction in real-time," *IEEE Intelligent Vehicles Symposium*, pp. 963–968, 2011.

[64] B. Kitt, F. Moosmann, and C. Stiller, "Moving on to dynamic environments: Visual odometry using feature classification," *IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems*, pp. 5551–5556, 2010.

[65] D. Scaramuzza and F. Fraundorfer, "Visual Odometry [Tutorial]," *IEEE Robotics & Automation Magazine*, vol. 18, no. 4, pp. 80–92, 2011.

[66] J.-P. Tardif, Y. Pavlidis, and K. Daniilidis, "Monocular visual odometry in urban environments using an omnidirectional camera," *International Conference on Intelligent Robots and Systems, 2008*, pp. 2531–2538, 2008.

[67] P. Beardsley, P. Torr, and A. Zisserman, "3D Model Aquisition from Extended Image Sequence," in *European Conference on Computer Vision*, vol. 2. Springer, 1996, pp. 683–695.

[68] G. Wang, "Robust Structure and Motion Factorization of Non-Rigid Objects," *Frontiers in Robotics and AI*, vol. 2, pp. 1–9, 2015.

[69] S. Im, H. Ha, G. Choe, H.-G. Jeon, K. Joo, and I. S. Kweon, "High Quality Structure from Small Motion for Rolling Shutter Cameras," in *IEEE International Conference on Computer Vision*. IEEE, 2015, pp. 837–845.

[70] J. Hedborg, E. Ringaby, P. E. Forssén, and M. Felsberg, "Structure and motion estimation from rolling shutter video," *Proceedings of the IEEE International Conference on Computer Vision*, pp. 17–23, 2011.

[71] Z. Cui, N. Jiang, C. Tang, and P. Tan, "Linear Global Translation Estimation with Feature Tracks," *British Machine Vision Conference*, no. 3, p. 2014, 2015.

[72] Z. Cui and P. Tan, "Global Structure-from-Motion by Similarity Averaging," in *IEEE International Conference on Computer Vision*, vol. 11-18. IEEE, 2015, pp. 864–872.

[73] N. Jiang, Z. Cui, and P. Tan, "A Global Linear Method for Camera Pose Registration," in *IEEE International Conference on Computer Vision*. IEEE, 2013, pp. 481–488.

[74] V. Govindu, "Combining two-view constraints for motion estimation," in *Computer Vision and Pattern Recognition*, vol. 2. IEEE, 2001, pp. 218–225.

[75] D. Martinec and T. Pajdla, "Robust Rotation and Translation Estimation in Multiview Reconstruction," in *Conference on Computer Vision and Pattern Recognition*, vol. 27, no. 5. IEEE, 2007, pp. 1–8.

[76] O. Enqvist, F. Kahl, and C. Olsson, "Non-sequential structure from motion," in *IEEE International Conference on Computer Vision Workshops*. IEEE, 2011, pp. 264–271.

[77] M. Arie-Nachimson, S. Z. Kovalsky, I. Kemelmacher-Shlizerman, A. Singer, and R. Basri, "Global Motion Estimation from Point Matches," in *International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission*. IEEE, 2012, pp. 81–88.

[78] V. M. Govindu, "Lie-algebraic averaging for globally consistent motion estimation," in *Computer Vision and Pattern Recognition*, vol. 1, 2004, pp. 1–8.

[79] P. Moulon, P. Monasse, and R. Marlet, "Global Fusion of Relative Motions for Robust, Accurate and Scalable Structure from Motion," in *IEEE International Conference on Computer Vision*. IEEE, 2013, pp. 3248–3255.

[80] M. Farenzena, A. Fusiello, and R. Gherardi, "Structure-and-motion pipeline on a hierarchical cluster tree," in *International Conference on Computer Vision Workshops*. IEEE, 2009, pp. 1489–1496.

[81] M. Havlena, A. Torii, and T. Pajdla, "Efficient structure from motion by graph optimization," in *European Conference on Computer Vision*. Springer Berlin Heidelberg, 2010, pp. 100–113.

[82] D. Nister and H. Stewenius, "Scalable Recognition with a Vocabulary Tree," in *Computer Vision and Pattern Recognition*, vol. 2. IEEE, pp. 2161–2168.

[83] H. Jégou, M. Douze, and C. Schmid, "Product Quantization for Nearest Neighbor Search Herve," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 1, pp. 117–128, 2011.

[84] K. Schindler, W. Hartmann, and M. Havlena, "Recent developments in large-scale tie-point matching," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 115, pp. 47–62, 2016.

[85] L. Svarm, Z. Simayijiang, O. Enqvist, and C. Olsson, "Point Track Creation in Unordered Image Collections Using Gomory-Hu Trees," *International Conference on Pattern Recognition*, pp. 2116–2119, 2012.

[86] N. Snavely, R. Garg, S. M. Seitz, and R. Szeliski, "Finding paths through the world's photos," *Association for Computing Machinery Transactions on Graphics Transactions on Graphics*, vol. 27, no. 3, p. 15, 2008.

[87] C. Häne, L. Heng, G. H. Lee, A. Sizov, and M. Pollefeys, "Real-time direct dense matching on fisheye images using plane-sweeping stereo," in *International Conference on 3D Vision*. IEEE, 2015, pp. 57–64.

[88] A. Wendel, C. Hoppe, H. Bischof, and F. Leberl, "Automatic Fusion of Partial Reconstructions," *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, pp. 1–3, 2012.

[89] Y. Furukawa and J. Ponce, "Accurate , Dense , and Robust Multi-View Stereopsis," *Conference on Computer Vision and Pattern Recognition*, vol. 32, no. 8, pp. 1362–1376, 2010.

[90] N. Snavely, S. M. Seitz, and R. Szeliski, "Skeletal sets for efficient structure from motion," in *Computer Vision and Pattern Recognition*. IEEE, 2008, p. 2.

[91] C. Sweeney, T. Sattler, T. Hollerer, M. Turk, and M. Pollefeys, "Optimizing the viewing graph for structure-from-motion," in *Proceedings of the IEEE International Conference on Computer Vision*, vol. 11. IEEE, 2015, pp. 801–809.

[92] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM : Real-Time Single Camera SLAM," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 29, no. 6, pp. 1–16, 2007.

[93] T. Whelan, S. Leutenegger, R. F. Salas-moreno, B. Glocker, and A. J. Davison, "ElasticFusion : Dense SLAM Without A Pose Graph," in *Robotics: Science and Systems*, vol. 11, 2015.

[94] H. Lim, J. Lim, and H. J. Kim, "Real-Time 6-DOF Monocular Visual SLAM in a Large-Scale Environment," *International Conference on Robotics and Automation*, pp. 1532–1539, 2014.

[95] S. Lynen, T. Sattler, M. Bosse, J. A. Hesch, M. Pollefeys, and R. Siegwart, "Get out of my lab: Large-scale, real-time visual-intertial localization," in *Robotics: Science and Systems*, 2015, pp. 37–46.

[96] S. Rusinkiewicz and M. Levoy, "QSplat," in *Computer Graphics and Interactive Techniques*. ACM Press/Addison-Wesley Publishing Co., 2000, pp. 343–352.

[97] A. Locher and L. V. Gool, "Progressive Prioritized Multi-view Stereo," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3244–3252, 2016.

[98] M. Grundmann, V. Kwatra, and I. Essa, "Auto-directed video stabilization with robust L1 optimal camera paths," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 225–232, 2011.

[99] A. Censi, A. Fusiello, and V. Roberto, "Image stabilization by features tracking," *Proceedings - International Conference on Image Analysis and Processing*, pp. 665–669, 1999.

[100] M. a. Fischler and R. C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applicatlons to Image Analysis and Automated Cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381 – 395, 1981.

[101] A. Bak, S. Bouchafa, and D. Aubert, "Detection of independently moving objects through stereo vision and ego-motion extraction," in *Intelligent Vehicles Symposium*. IEEE, 2010, pp. 863–870.

[102] O. K. Hamilton, T. P. Breckon, X. Bai, and S. I. Kamata, "A foreground object based quantitative assessment of dense stereo approaches for use in automotive environments," in *International Conference on Image Processing*. IEEE, 2013, pp. 418–422.

[103] A. Wedel, A. Meißner, C. Rabe, U. Franke, and D. Cremers, "Detection and segmentation of independently moving objects from dense scene flow," in *Lecture Notes in Computer Science*, 2009, vol. 5681, pp. 14–27.

[104] J.-Y. Kao, D. Tian, H. Mansour, A. Vetro, and A. Ortega, "Moving object segmentation using depth and optical flow in car driving sequences," in *IEEE International Conference on Image Processing*. IEEE, 2016, pp. 11–15.

[105] R. Haeusler and R. Klette, "Analysis of KITTI Data for Stereo Analysis with Stereo Confidence Measures," in *European Conference on Computer Vision*. Springer, 2012, pp. 158–167.

[106] V. Kolmogorov and R. Zabih, "Computing visual correspondence with occlusions using graph cuts," in *International Conference on Computer Vision*, vol. 2, no. 1. IEEE, 2001, pp. 508–515.

[107] C. Lei, J. Selzer, and Y. H. Yang, "Region-tree based stereo using dynamic programming optimization," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 2378–2385, 2006.

[108] J. Zbontar and Y. LeCun, "Computing the stereo matching cost with a convolutional neural network," in *Computer Vision and Pattern Recognition*. IEEE, 2015, pp. 1592–1599.

[109] A. Shaked and L. Wolf, "Improved Stereo Matching with Constant Highway Networks and Reflective Confidence Learning," *arXiv preprint arXiv:1701.00165*, 2016.

[110] Velodyne, "Velodyne Lidar," 2017. [Online]. Available: http://velodynelidar.com/

[111] P. Besl and N. McKay, "A Method for Registration of 3-D Shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[112] H. Kruegle, *CCTV Surveillance: Video Practices and Technology*, 2nd ed. Butterworth-Heinemann, 2011.

[113] F. Navarro, F. J. Seron, and D. Gutierrez, "Motion blur rendering: State of the art," *Computer Graphics Forum*, vol. 30, no. 1, pp. 3–26, 2011.

[114] G. Klein and D. D. Murray, "Parallel tracking and mapping for small AR workspaces," in *International Symposium on Mixed and Augmented Reality*, vol. 07, no. 3. Ieee, 2007, pp. 1–10.

[115] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," in *Proceedings AAAI National Conference on Artificial Intelligence*, 1999.

[116] S. Gauglitz, T. Höllerer, and M. Turk, "Evaluation of interest point detectors and feature descriptors for visual tracking," *International Journal of Computer Vision*, vol. 94, no. 3, pp. 335–360, 2011.

[117] R. I. Hartley and P. Sturm, "Triangulation," *Computer Vision and Image Understanding*, vol. 68, no. 2, pp. 146–157, 1997.

[118] J. Nurmi, E. S. Lohan, S. Sand, and H. Hurskainen, *GALILEO Positioning Technology*, ser. Signals and Communication Technology, J. Nurmi, E. S. Lohan, S. Sand, and H. Hurskainen, Eds. Dordrecht: Springer Netherlands, 2015, vol. 182.

[119] M. Martin-Neira, P. Colmenarejo, G. Ruffini, and C. Serra, "Altimetry precision of 1 cm over a pond using the wide-lane carrier phase of GPS reflected signals," *Canadian Journal of Remote Sensing*, vol. 28, no. 3, pp. 394–403, 2002.

[120] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 Year, 1000km: The Oxford RobotCar Dataset," *The International Journal of Robotics Research*, vol. 3, no. 2014, 2014.

[121] S. Jianbo and C. Tomasi, "Good Features to Track," in *Computer Vision and Pattern Recognition*. IEEE, 1994, pp. 593–600.

[122] B. D. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision," *International Joint Conference on Artificial Intelligence*, vol. 130, pp. 674–679, 1981.

[123] P. Cavestany, A. L. Rodriguez, H. Martinez-Barbera, and T. P. Breckon, "Improved 3D sparse maps for high-performance SFM with low-cost omnidirectional robots," in *International Conference on Image Processing*. IEEE, 2015, pp. 4927–4931.

[124] M. I. A. Lourakis and A. A. Argyros, "SBA: A Software Package for Generic Sparse Bundle Adjustment," *ACM Transactions on Mathematical Software*, vol. 36, no. 1, pp. 1–30, 2009.

[125] A. Delaunoy and M. Pollefeys, "Photometric Bundle Adjustment for Dense Multi-view 3D Modeling," in *Conference on Computer Vision and Pattern Recognition*. IEEE, 2014, pp. 1486–1493.

[126] M. Lhuillier, "Incremental fusion of structure-from-motion and GPS using constrained bundle adjustments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 12, pp. 2489–2495, 2012.

[127] R. Toldo, R. Gherardi, M. Farenzena, and A. Fusiello, "Hierarchical structure-and-motion recovery from uncalibrated images," *Computer Vision and Image Understanding*, vol. 140, pp. 127–143, 2015.

[128] G. Zhang, J. Jia, T.-T. Wong, and H. Bao, "Recovering consistent video depth maps via bundle optimization," in *Conference on Computer Vision and Pattern Recognition*, vol. 20, ACM New York, NY, USA. IEEE, 2008, pp. 1–8.

[129] Q. Luong and O. Faugeras, "Camera calibration, scene motion and structure recovery from point correspondences and fundamental matrices," *International Journal of Computer Vision*, vol. 22, no. 3, pp. 261–289, 1997.

[130] R. Y. Tsai, "A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses," *IEEE Journal on Robotics and Automation*, vol. 3, no. 4, pp. 323–344, 1987.

[131] Z. Zhang, "Flexible camera calibration by viewing a plane from unknown orientations," in *IEEE International Conference on Computer Vision*, vol. 1. IEEE, 1999, pp. 666–673.

[132] M. Vo, S. G. Narasimhan, and Y. Sheikh, "Spatiotemporal Bundle Adjustment for Dynamic 3D Reconstruction," in *Computer Vision and Pattern Recognition*, 2016, pp. 1710–1718.

[133] F. Fraundorfer, D. Scaramuzza, and M. Pollefeys, "A constricted bundle adjustment parameterization for relative scale estimation in visual odometry," in *International Conference on Robotics and Automation*. IEEE, 2010, pp. 1899–1904.

[134] K. Konolige, M. Agrawal, and J. Solà, "Large-scale visual odometry for rough terrain," *Robotics research*, vol. 66, pp. 201–212, 2010.

[135] J.-P. Tardif, M. George, M. Laverne, A. Kelly, and A. Stentz, "A new approach to vision-aided inertial navigation," in *International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 4161–4168.

[136] O. Mora, J. J. Mallorqui, and A. Broquetas, "Linear and nonlinear terrain deformation maps from a reduced set of interferometric SAR images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 41, no. 10, pp. 2243–2253, 2003.

[137] J. Hariyono, V.-D. Hoang, and K.-H. Jo, "Moving object localization using optical flow for pedestrian detection from a moving vehicle." *The Scientific World Journal*, 2014.

[138] G. Farneb, "Two-Frame Motion Estimation Based on Polynomial Expansion," *Lecture Notes in Computer Science*, vol. 2749, no. 1, pp. 363–370, 2003.

[139] L. Heng, B. Li, and M. Pollefeys, "CamOdoCal: Automatic intrinsic and extrinsic calibration of a rig with multiple generic cameras and odometry," *IEEE International Conference on Intelligent Robots and Systems*, pp. 1793–1800, 2013.

[140] I. Nourbakhsh and R. Siegwart, *Introduction to Autonomous Mobile Robots*. MIT Press, 2004.

[141] S. Agarwal and K. Mierle, "Ceres Solver," 2016. [Online]. Available: http://ceres-solver.org

[142] F. Zhang, *The Schur Complement and Its Applications*, ser. Numerical Methods and Algorithms, F. Zhang, Ed. New York: Springer US, 2005, vol. 4.

[143] R. Tylecek and R. Sara, "Refinement of Surface Mesh for Accurate Multi-View Reconstruction," *International Journal of Virtual Reality*, vol. 9, no. 1, pp. 45–54, 2010.

Appendix A

Supplementary Material

A.1 Further Stereo Results

The following section contains further examples of normalised dense disparity maps to illustrate relative quality.

Figure A.1: Dense stereo disparity maps generated using the KITTI data. Algorithms used, top to bottom: BM, SGBM, NoMD, Cross, AdaptDP, ELAS.

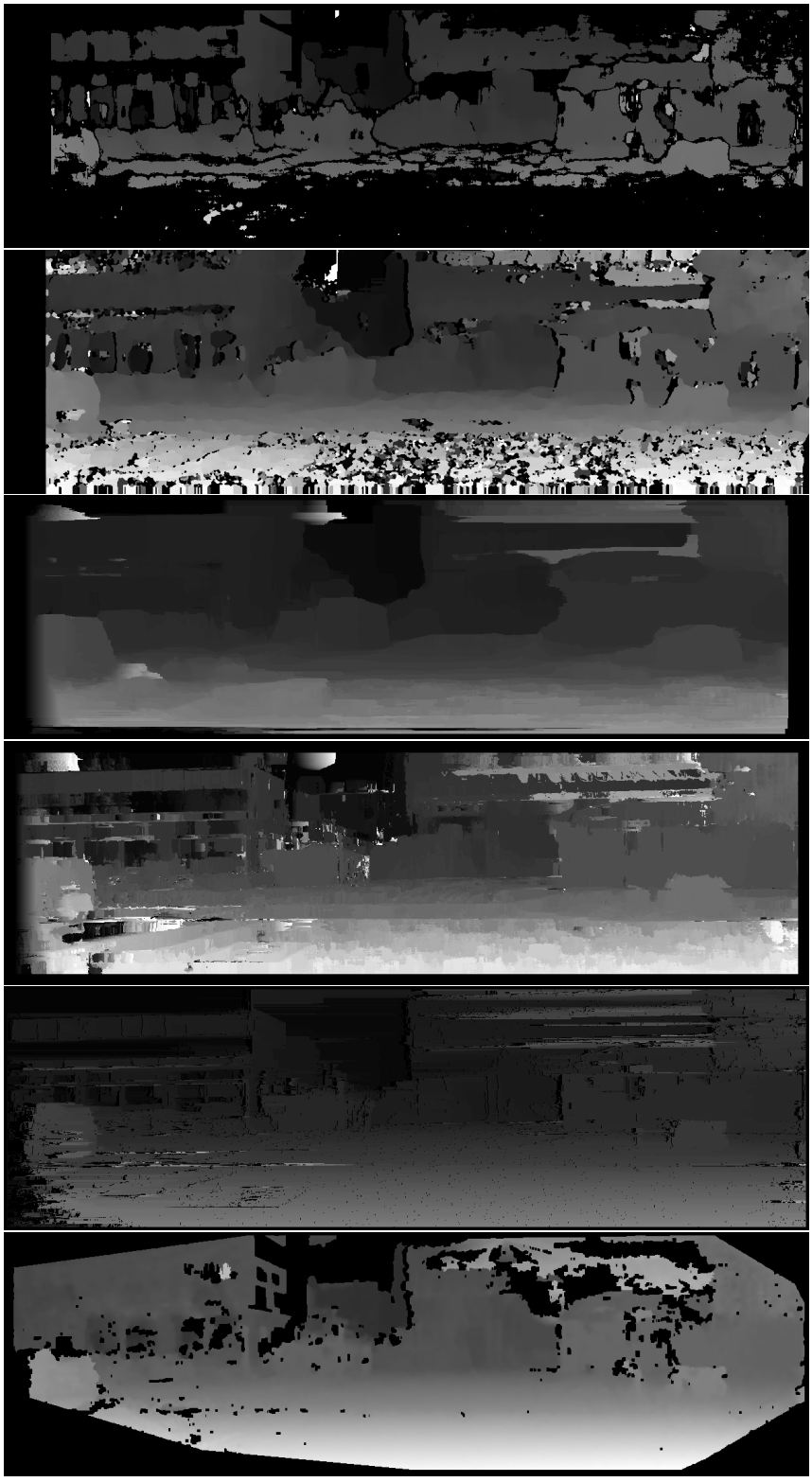

Figure A.2: Dense stereo disparity maps generated using the KITTI data. Algorithms used, top to bottom: BM, SGBM, NoMD, Cross, AdaptDP, ELAS.

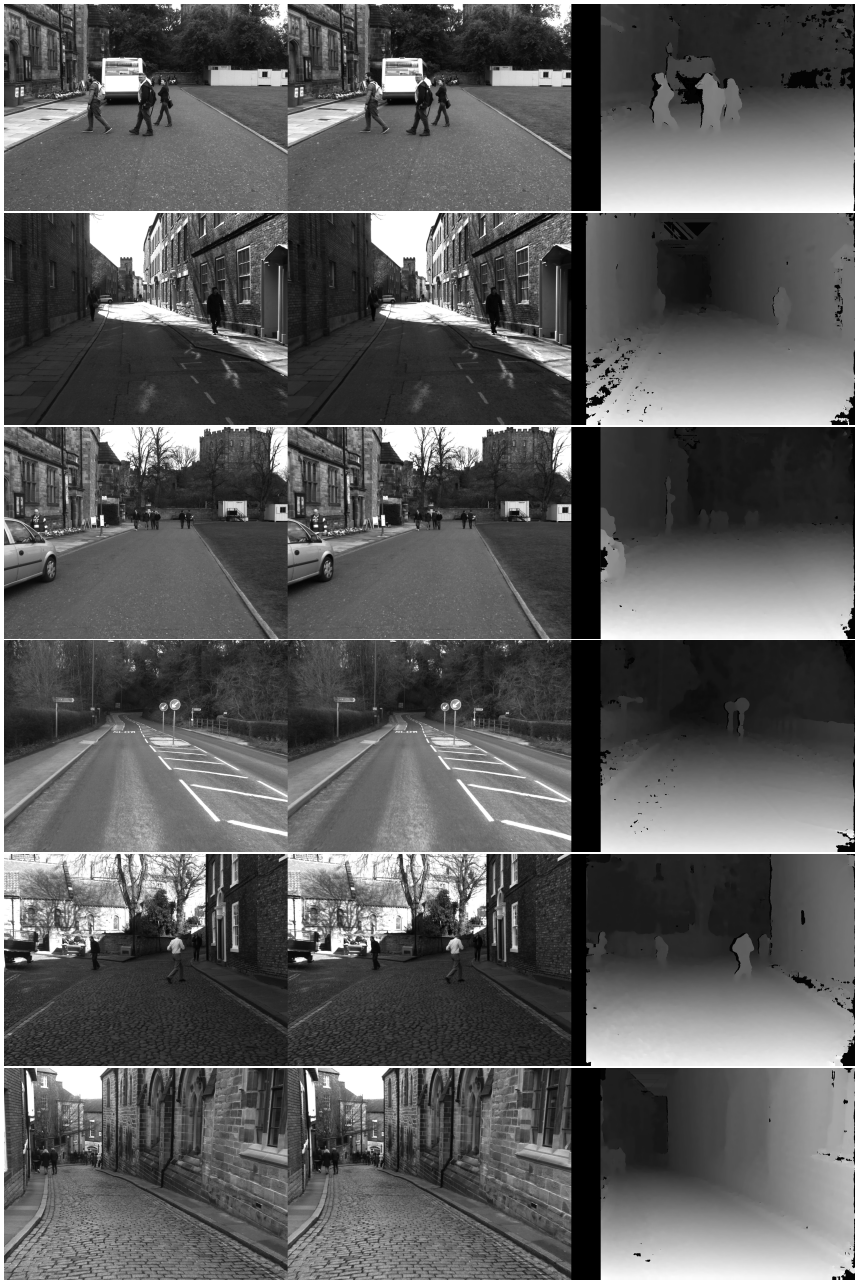

Figure A.3: Left and middle: Stereo input images from our data set. Right: Dense disparity map created with SGBM.

A.2 Further Dense Stereo Range Angle Maps

The following are further examples of the range angle maps from Section 3.3.

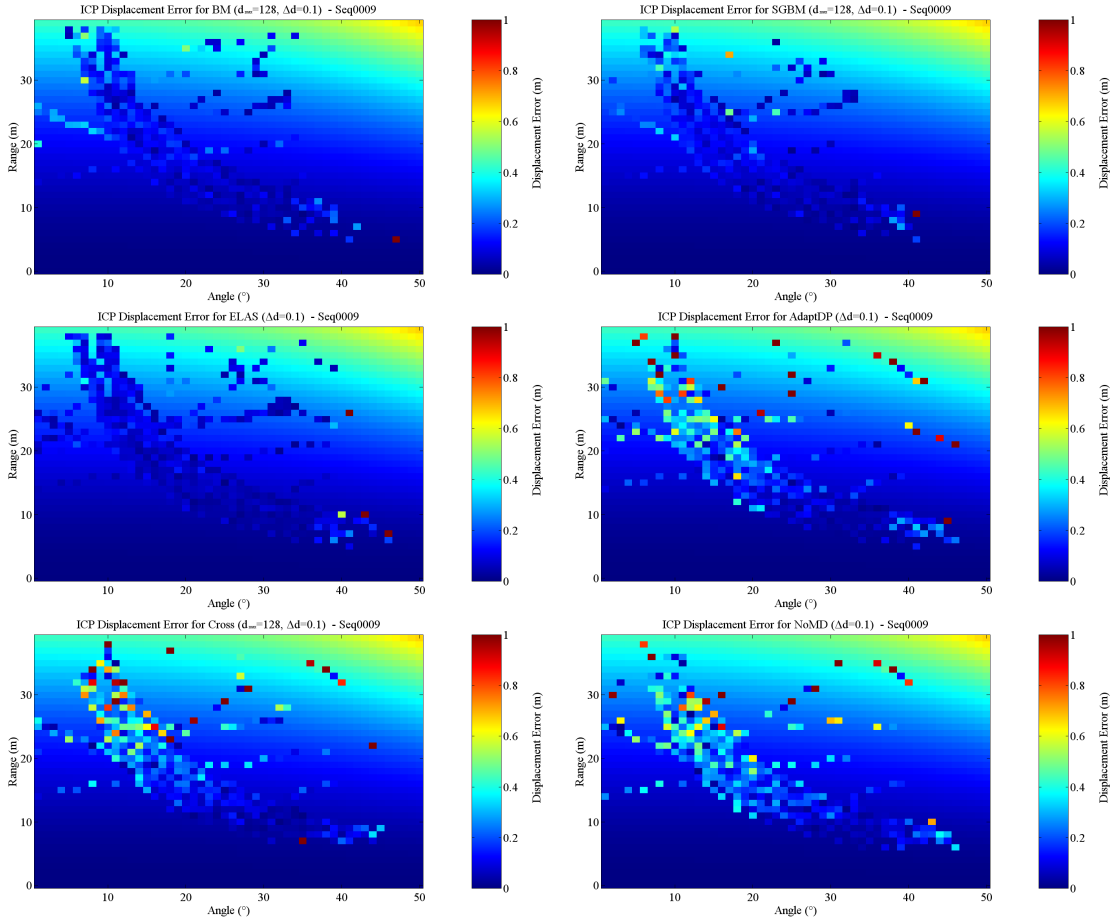

Figure A.4: Range-angle maps showing accuracy of dense stereo algorithms from KITTI image sequence 2011_09_26_drive_0009 using process outlined in Chapter 3.

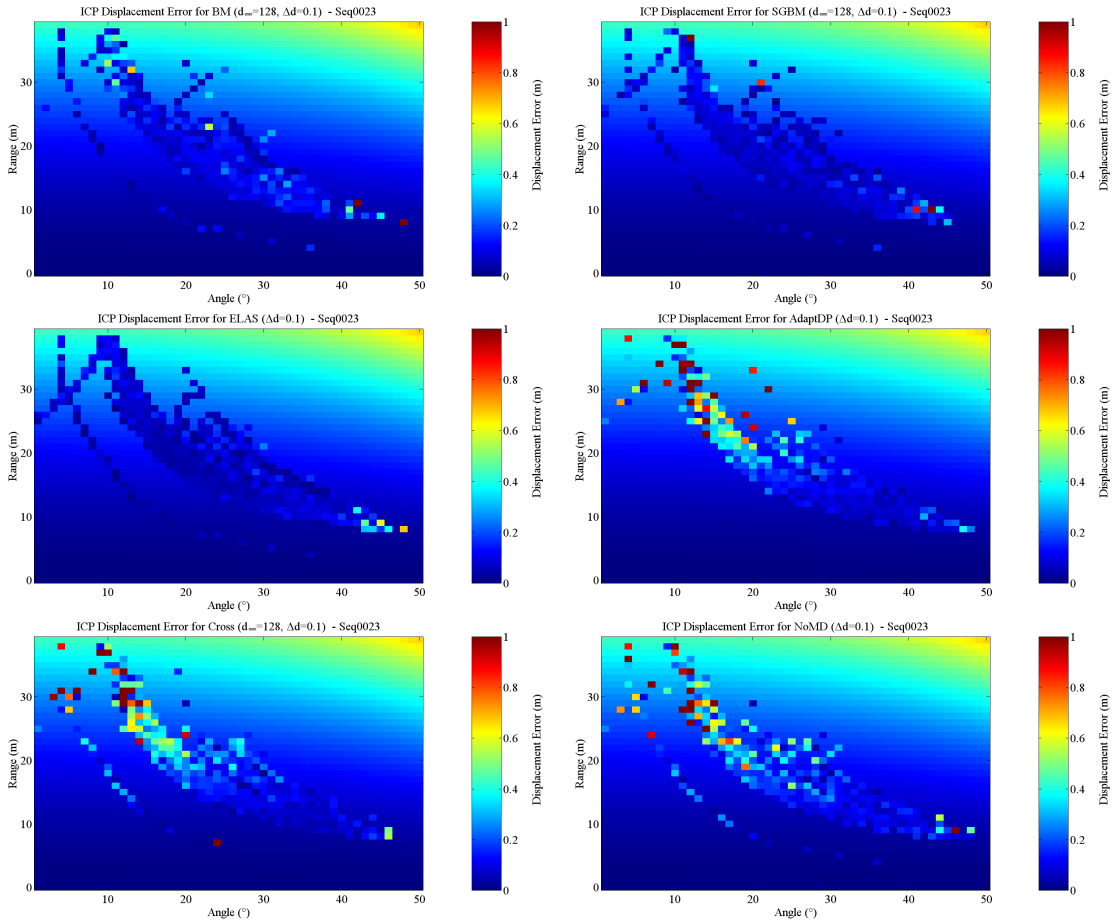

Figure A.5: Range-angle maps showing accuracy of dense stereo algorithms from KITTI image sequence 2011_09_26_drive_0023 using process outlined in Chapter 3.

A.3 Optical Flow Optimisaton Results

Full raw results from optical flow parameter tuning.

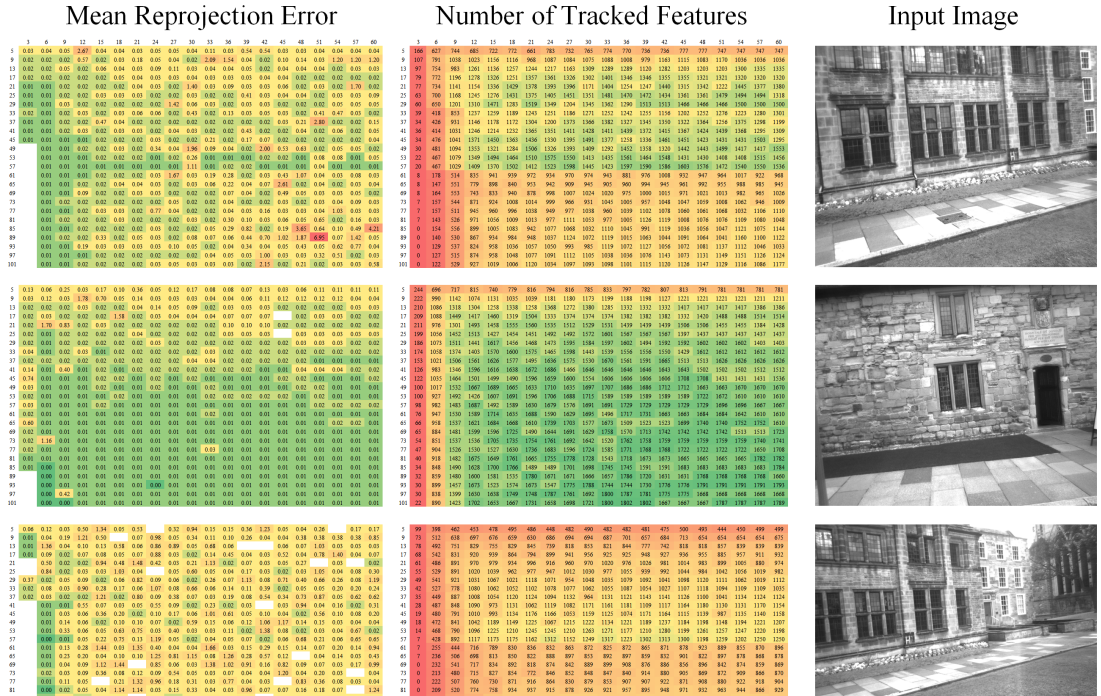

Figure A.6: Full results for the charts in Figure 4.8.

A.4 Dynamic Bundle Adjustment Windows Size and Filter Strategy Results

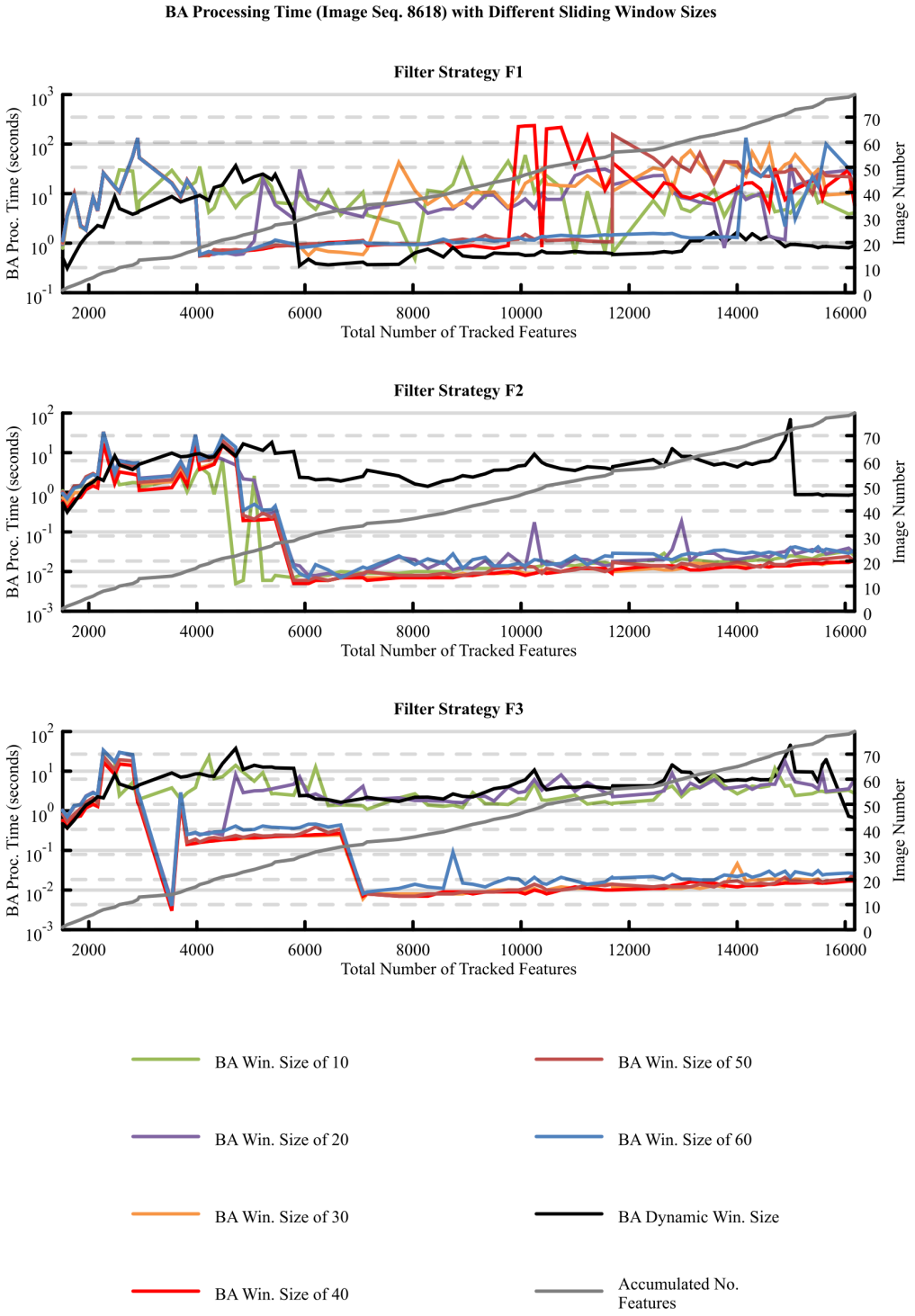

Figure A.7: Bundle adjustment processing times for different window sizes and filtering strategies. Sequence 1 in Section 4.2.4.7.

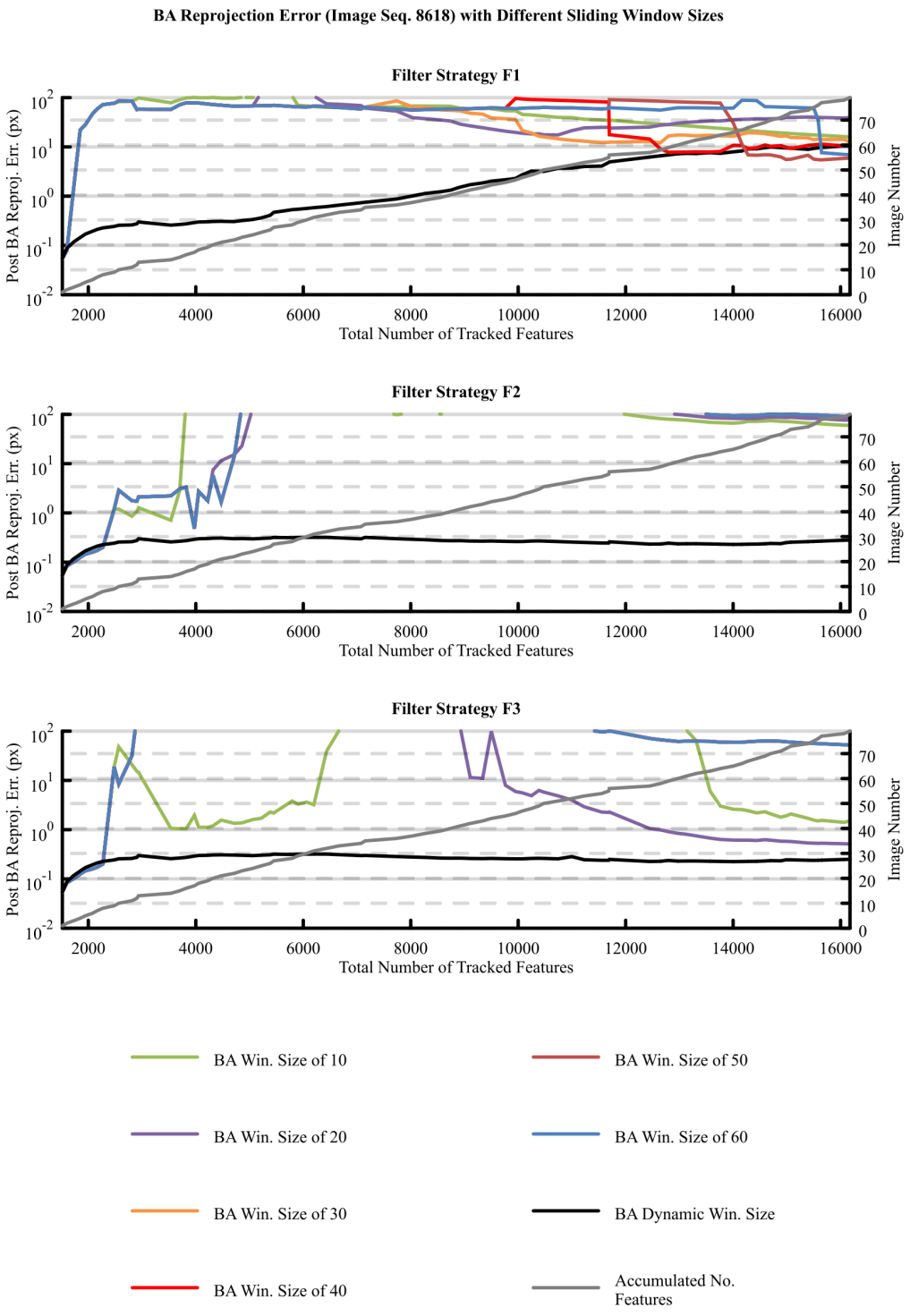

Figure A.8: Reprojection error post BA stage of the SfM process for different sized temporal windows and filter strategies. Sequence 1 in Section 4.2.4.7.

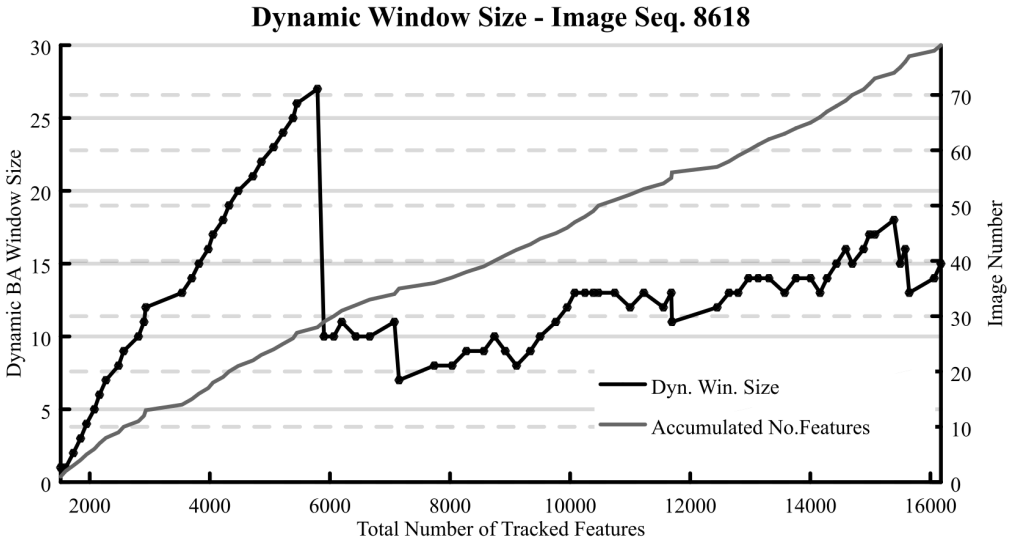

Figure A.9: Dynamic window sizes used for our bundle adjustment approach of Sequence 1 in Section 4.2.4.7.

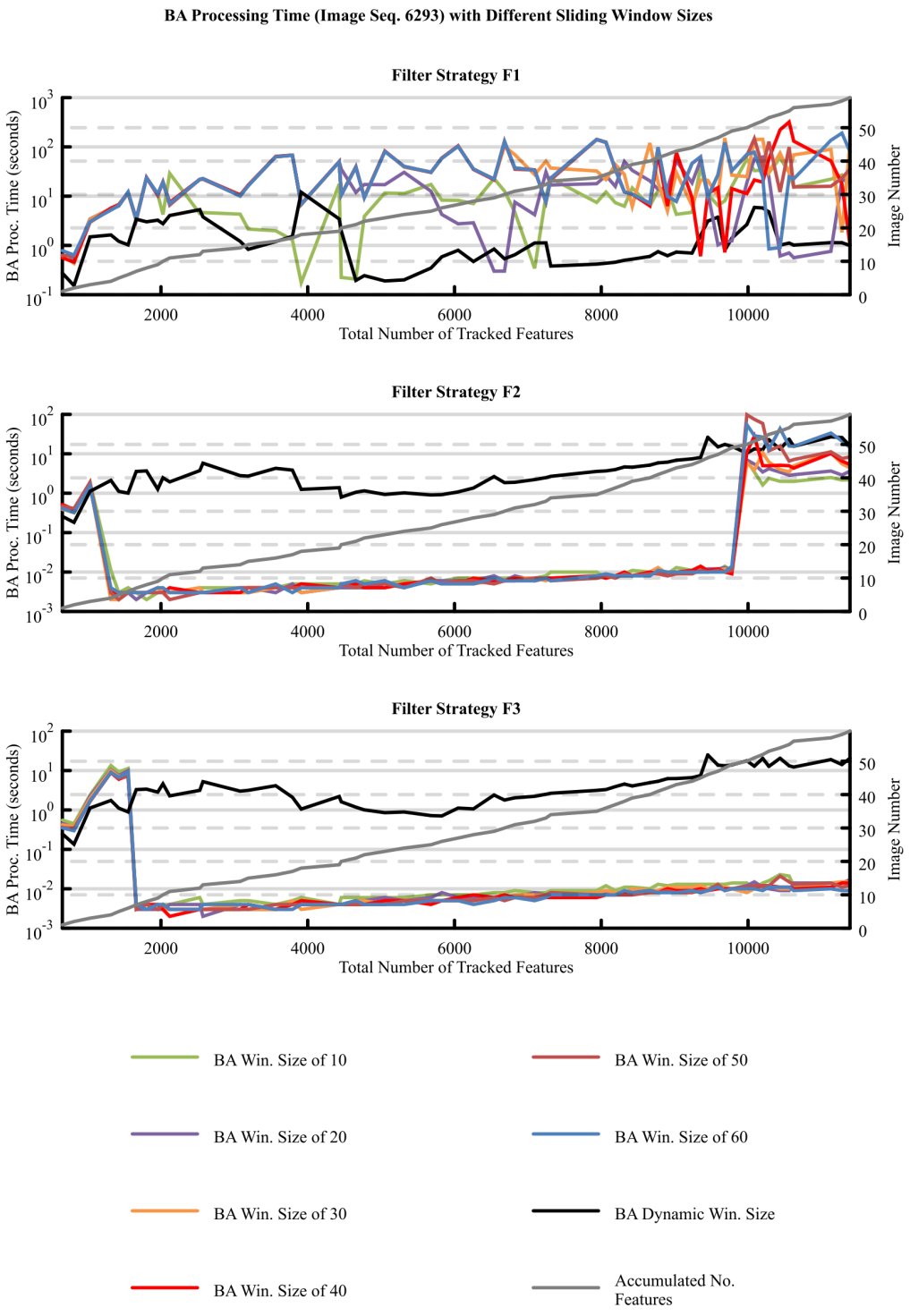

Figure A.10: Bundle adjustment processing times for different window sizes and filtering strategies. Sequence 4 in Section 4.2.4.7.

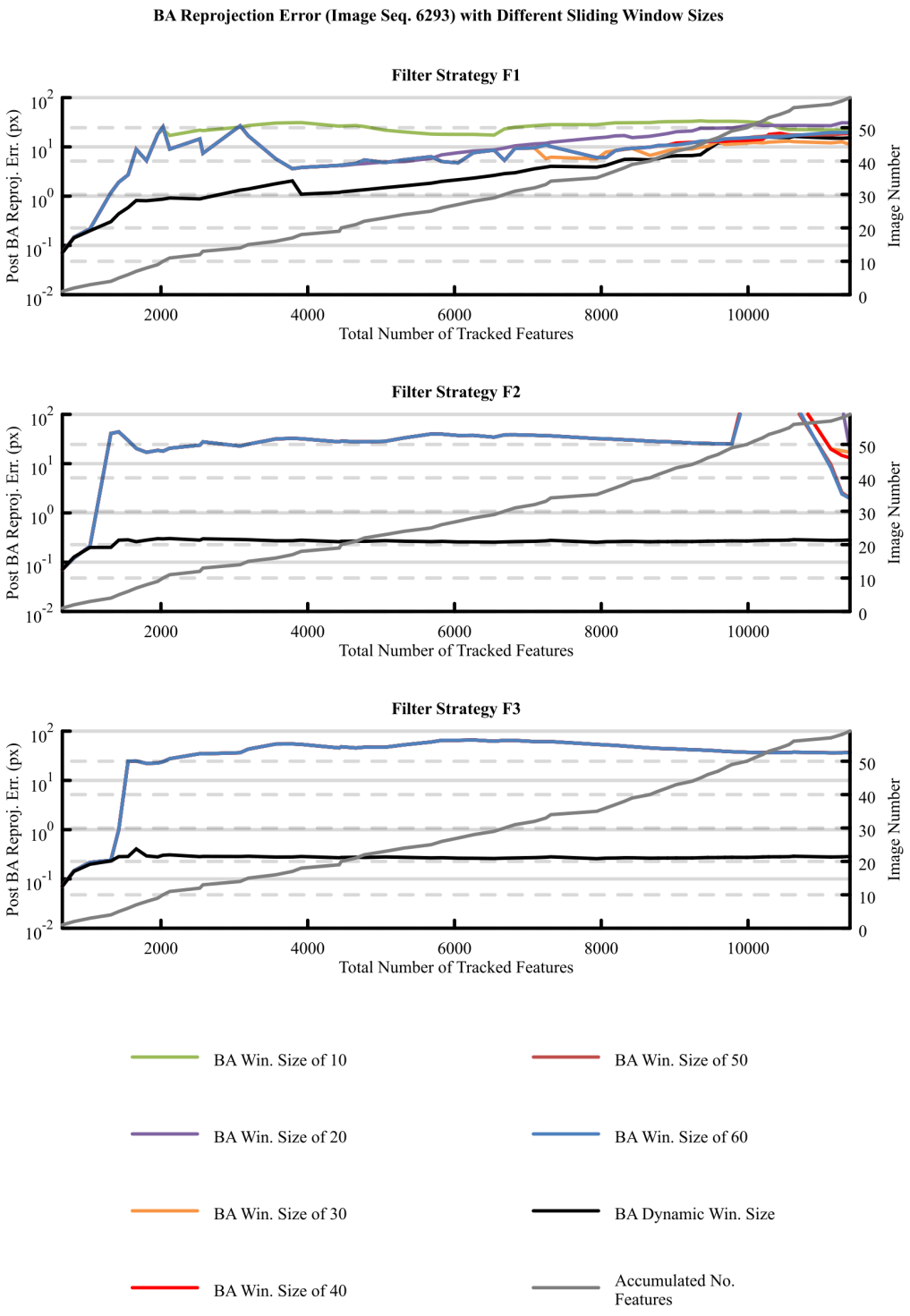

Figure A.11: Reprojection error post BA stage of the SfM process for different sized temporal windows and filter strategies. Sequence 4 in Section 4.2.4.7.

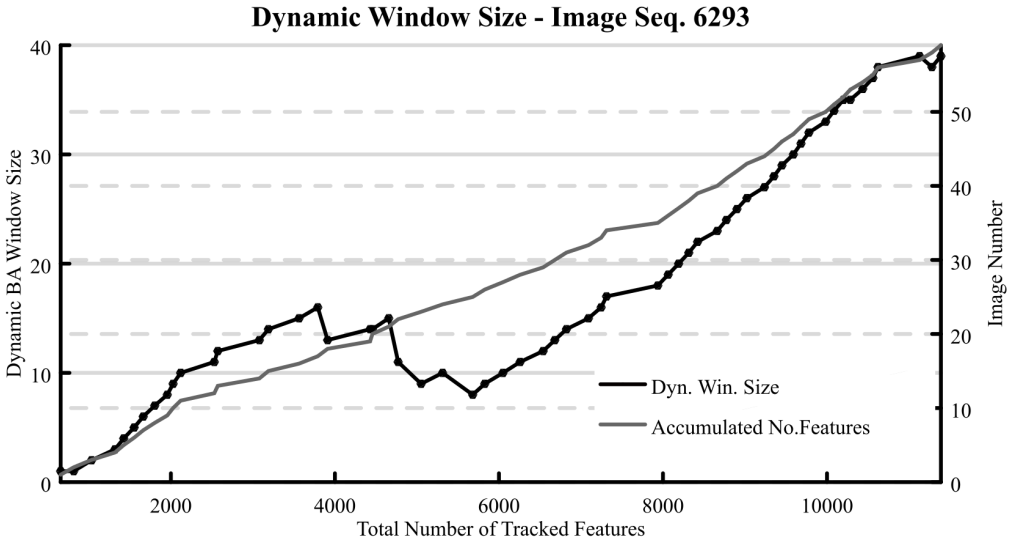

Figure A.12: Dynamic window sizes used for our bundle adjustment approach of Sequence 4 in Section 4.2.4.7.

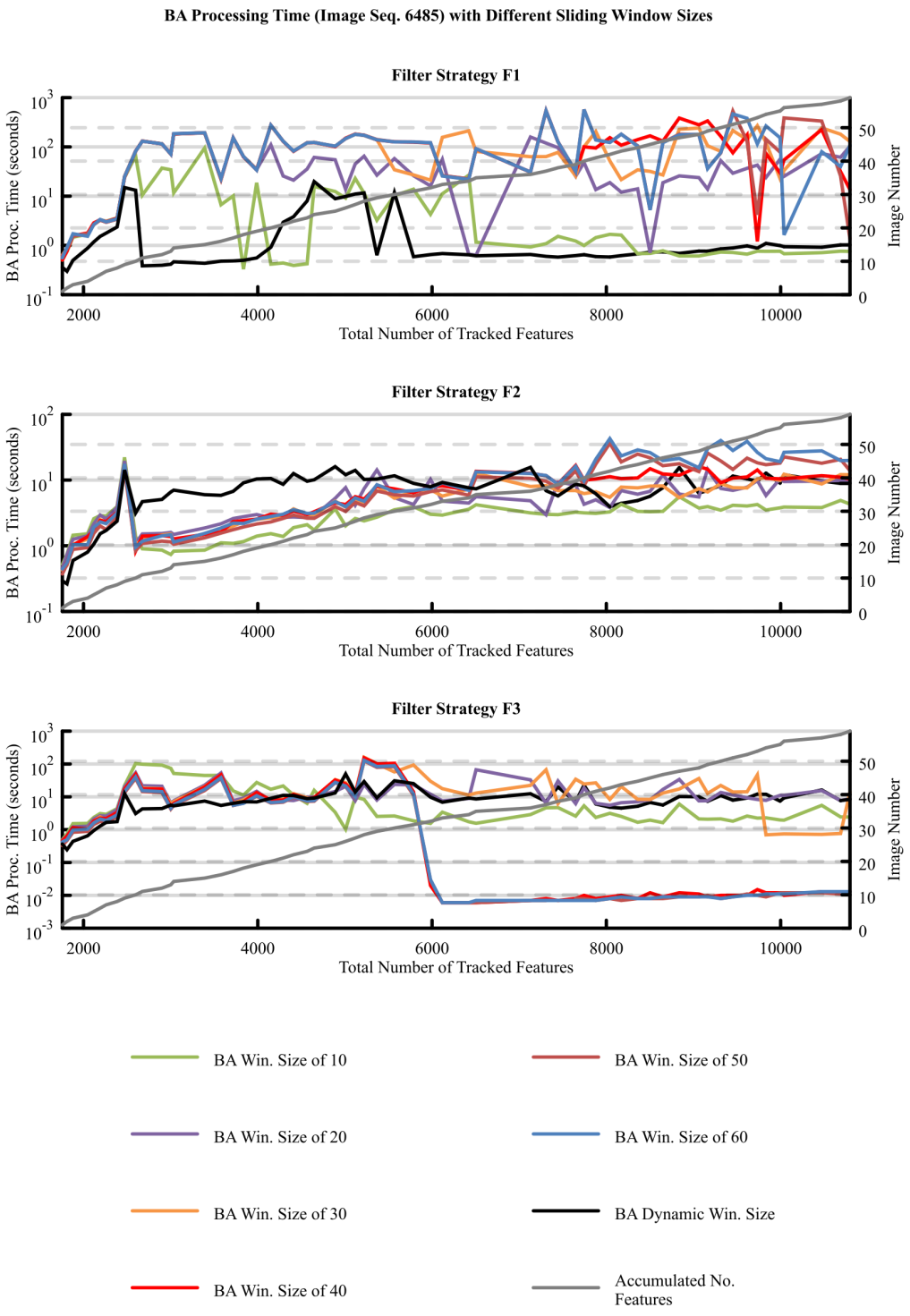

Figure A.13: Bundle adjustment processing times for different window sizes and filtering strategies. Sequence 6 in Section 4.2.4.7.

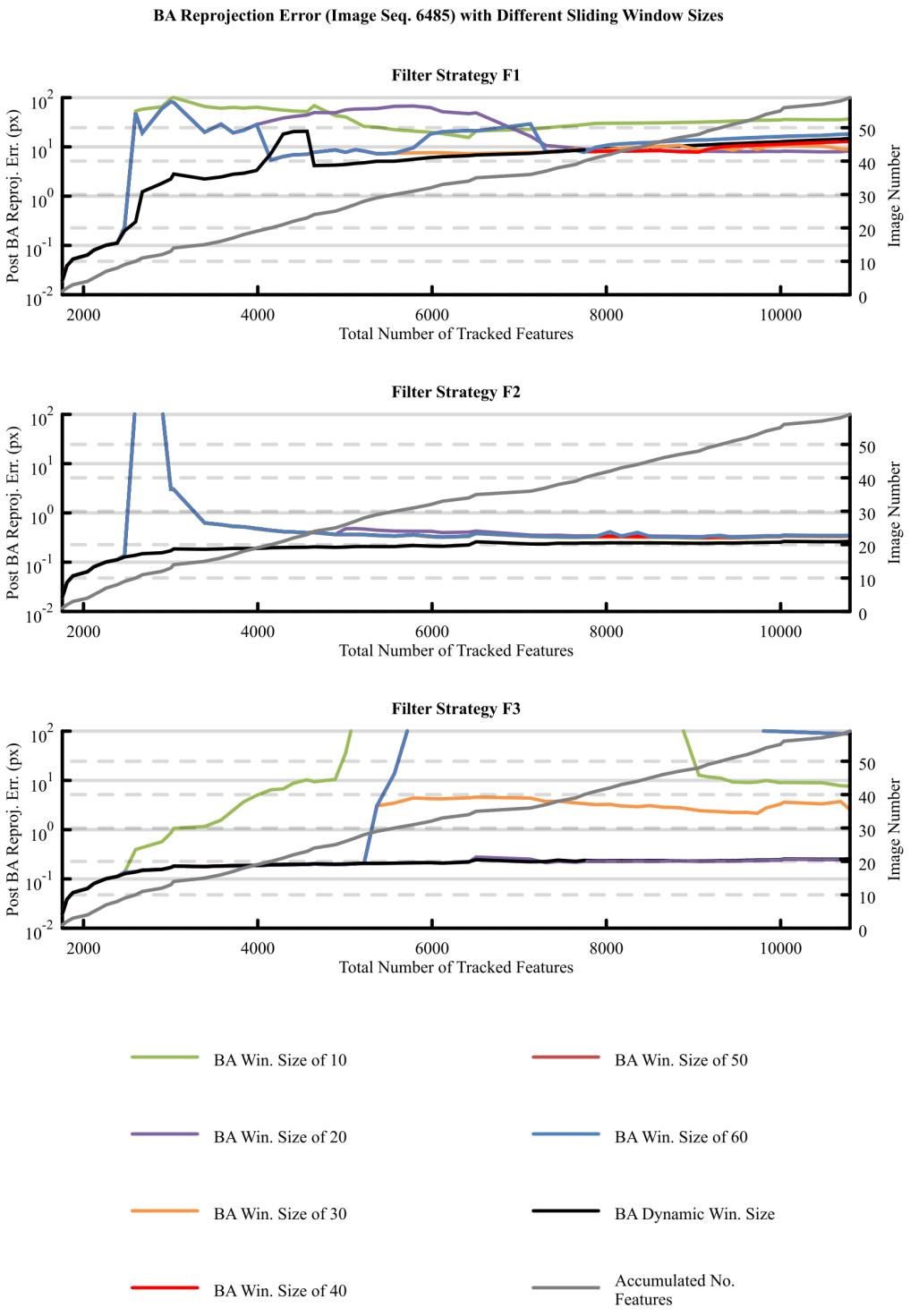

Figure A.14: Reprojection error post BA stage of the SfM process for different sized temporal windows and filter strategies. Sequence 6 in Section 4.2.4.7.

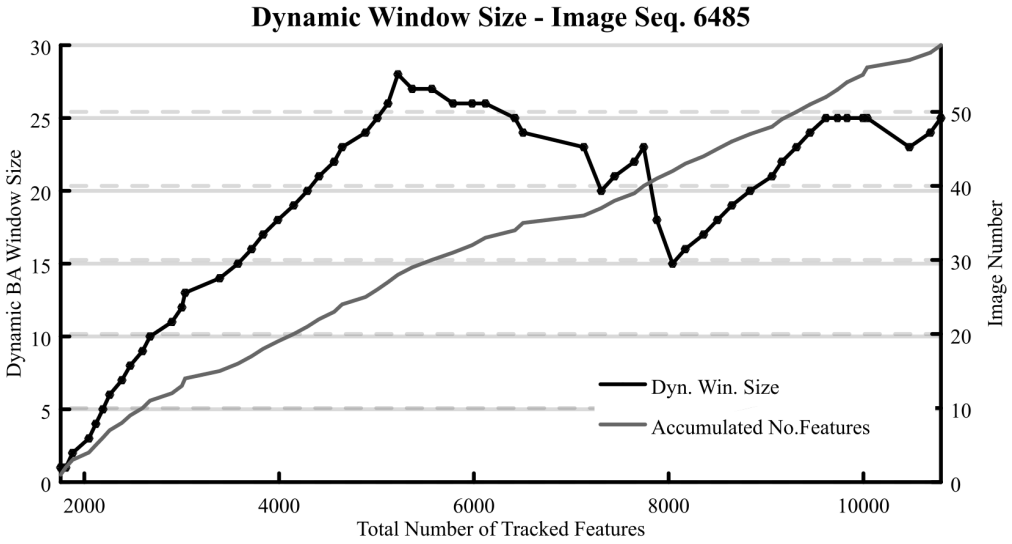

Figure A.15: Dynamic window sizes used for our bundle adjustment approach of Sequence 6 in Section 4.2.4.7.

A.5 Evolution of Data Capture Platform

For this project a custom data collection platform was designed, built and upgraded over several iterations.

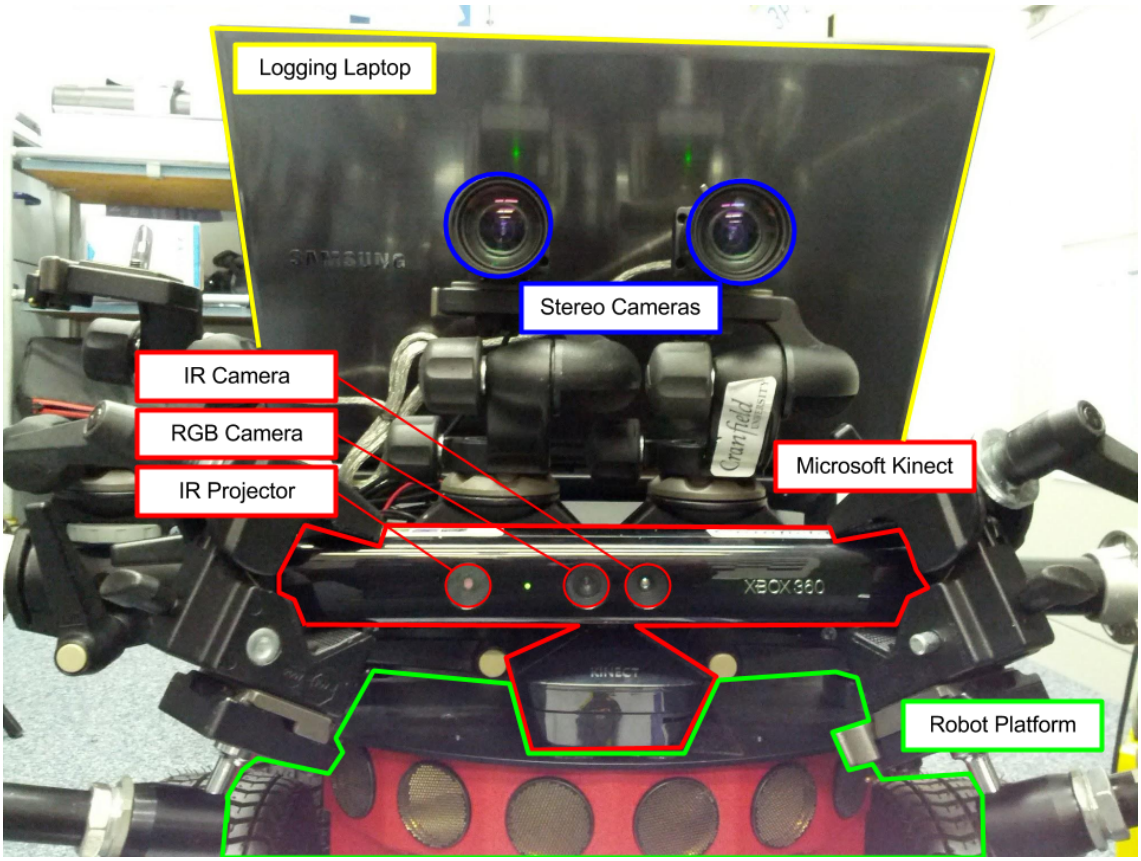

Figure A.16: Initial robot platform based configuration. Hardware components labelled. The short wheel base caused excessive roll and pitch when traversing even slightly rough ground, this manifested as image artifacts during the camera integration period due to a rolling shutter based CCD sensor.

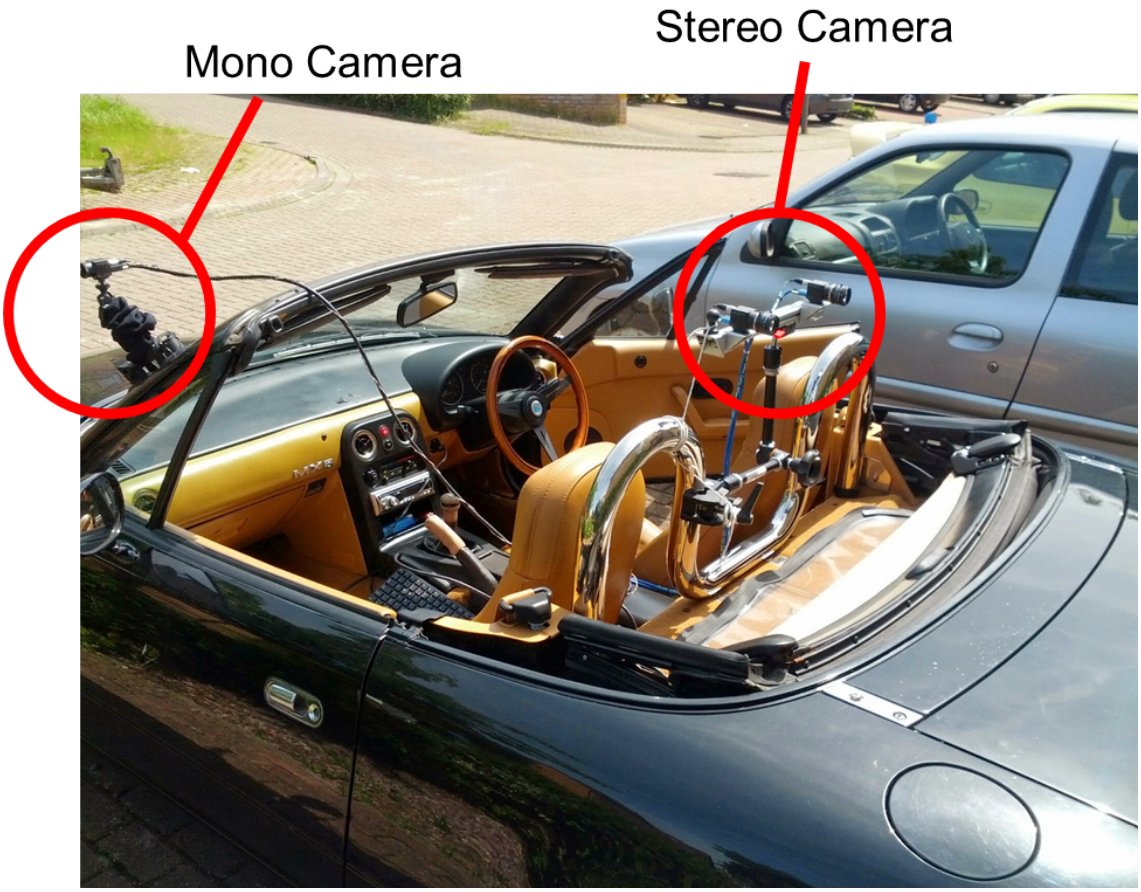

Figure A.17: The next data capture phase saw the addition of side-facing cameras and the mounting on a vehicle to reduce the effect rough surfaces. The cameras were also upgraded to global shutters to eliminate integration artifacts due to rolling shutter.

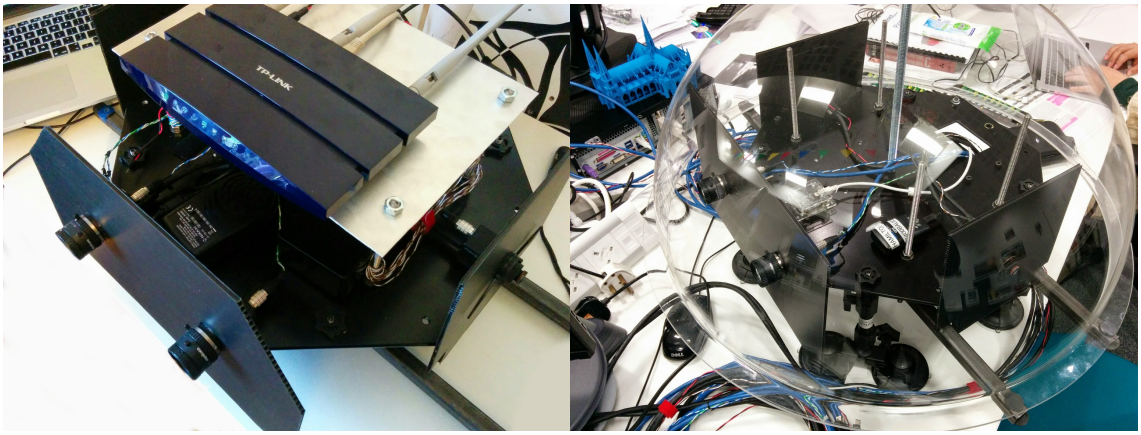

Figure A.18: Final version used in this work consists of a fully self-contained weather resistant logging platform capable of deployment on any vehicle. Image capture is from a forward facing stereo pair and two side facing mono cameras. Camera synchronisation is controlled via a 5v fixed pulse rate signal from an Arduino microcontroller. Final version mounted on vehicle can be seen in Figure 4.2.